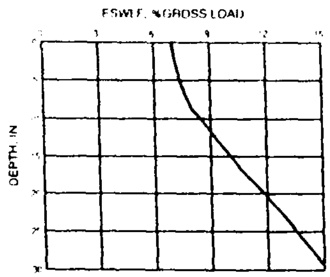
(2)

US Army Corps of Engineers

MISCELLANEOUS PAPER GL-92-18

A COMPUTER PROGRAM FOR THE CALCULATION OF THE EQUIVALENT SINGLE-WHEEL LOAD FACTOR



by

Carlos R. Gonzalez

Geotechnical Laboratory

DEPARTMENT OF THE ARMY
Waterways Experiment Station, Corps of Engineers
3909 Halls Ferry Road, Vicksburg, Mississippi 39180-6199

July 1992

Final Report

Prepared for DEPARTMENT OF THE ARMY
US Army Corps of Engineers
Washington, DC 20314-1000

and

DEPARTMENT OF THE AIR FORCE
Air Force Engineering and Services Center
Tyndall Air Force Base, Florida 32403-6001

GEOTECHNICAL LABORATORY

Best
Available
Copy

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
July 1992

3. REPORT TYPE AND DATES COVERED
Final report

4. TITLE AND SUBTITLE
A Computer Program for the Calculation of the Equivalent Single-Wheel Load Factor

5. FUNDING NUMBERS

6. AUTHOR(S)
Carlos R. Gonzalez

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
USAE Waterways Experiment Station
Geotechnical Laboratory
3909 Halls Ferry Road
Vicksburg, MS 39180-6199

8. PERFORMING ORGANIZATION REPORT NUMBER
Miscellaneous Paper GL-92-18

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
See reverse.

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
Available from National Technical Information Service, 5285 Port Royal Road, Springfield, VA 22161.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

A computer program for the calculation of the equivalent single-wheel load of any gear configuration using the Boussinesq solution for a circular load on a linear elastic half space is presented. Instructions as well as an example problem are included to demonstrate the use of the program. Recommendations on the implementation and user interface of engineer computer programs are also provided.

14. SUBJECT TERMS
See reverse.

15. NUMBER OF PAGES
63

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	

9. (Concluded).

US Army Corps of Engineers
Washington, DC 20314-1000

Air Force Engineering and Services Center
Tyndall Air Force Base, FL 32403-6001

14. (Concluded).

Boussinesq solution
Computer code
Computer program
Equivalent single-wheel load
Wheel load

PREFACE

The report herein was sponsored by Headquarters, US Army Corps of Engineers (USACE) and US Air Force under the effort "Design for Heavy-Weight Aircraft." The USACE Technical Monitor was Mr. Paige Johnson.

The computer program was developed at the US Army Engineer Waterways Experiment Station (WES) from May 1991 through September 1991 by the Pavement System Division (PSD) of the Geotechnical Laboratory (GL). This report was written by Mr. Carlos R. Gonzalez, PSD, under the supervision of Dr. George M. Hammitt II, Chief, PSD; and Mr. Jim Hall, Chief, System Analysis Branch. The work was performed under the general supervision of Dr. W. F. Marcuson III, Director, GL.

At the time of publication of this report, Director of WES was Dr. Robert W. Whalin. Commander and Deputy Director was COL Leonard G. Hassell, EN.

CONTENTS

	Page
PREFACE	1
CONVERSION FACTORS, NON-SI TO SI (METRIC) UNITS OF MEASUREMENT	3
PART I: INTRODUCTION	4
PART II: PROGRAMMING PHILOSOPHY AND SYSTEM REQUIREMENTS	7
PART III: USER INSTRUCTIONS	8
Running the Program	8
Entering Data	8
Example Problem	15
Additional Program Features	15
PART IV: CONCLUSIONS AND RECOMMENDATIONS	20
Conclusions	20
Recommendations	20
APPENDIX A: COMPUTER PROGRAM SOURCE CODE	A1

CONVERSION FACTORS, NON-SI TO SI (METRIC) UNITS OF MEASUREMENT

Non-SI units of measurement used in this report can be converted to SI (metric) units as follows:

Multiply	By	To Obtain
inches	2.54	centimetres
pounds (force)	4.448222	newtons
pounds (force) per square inch	6.894757	kilopascals
square inches	6.4516	square centimetres

A COMPUTER PROGRAM FOR THE CALCULATION OF THE EQUIVALENT SINGLE-WHEEL LOAD FACTOR

PART I: INTRODUCTION

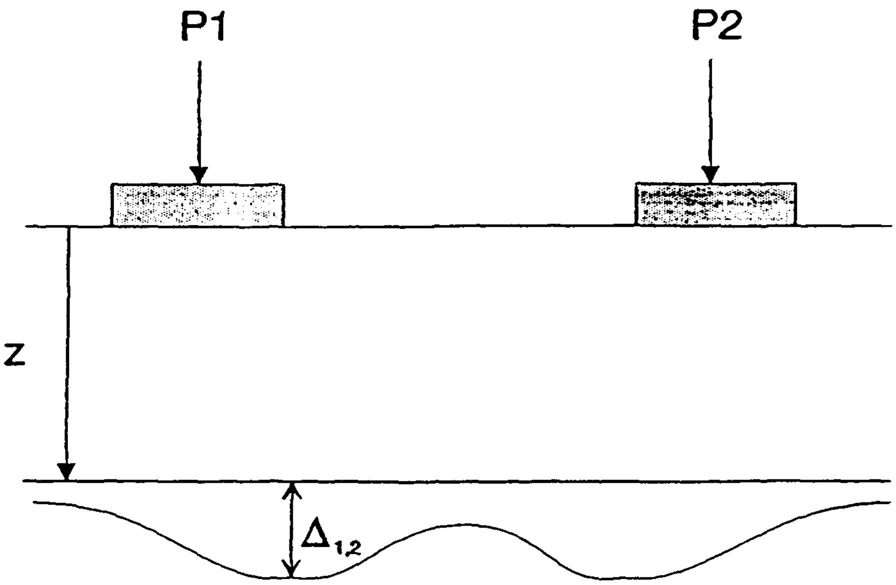
1. For many years the Corps of Engineers (CE) has been implementing the concept of an equivalent single-wheel load (ESWL) to represent the effects of multiple wheels in a vehicle gear assembly upon a pavement system. In the early years of the development of the CE pavement design procedure, the Boussinesq solution for the effects of circular loads in a linear elastic half space was selected to approximate the response of a pavement structure under wheel loads. Closed form solutions for computing the pavement response (stresses or strains) under multiple wheels were difficult and tedious to compute. Even today these solutions required considerable computer power. For this reason, the concept of converting a multi-wheel gear load to a single-wheel load that would produce the same effect at a given depth in a pavement system was introduced. This conversion or equivalency was based on vertical deflection at a given depth. Yoder and Witzak (1975)* and the BOEING Company (1963)** describe in detail the concept behind this equivalency. For the purpose of this report, it is sufficient to know that the ESWL is the load on a single wheel that will produce the same maximum vertical deflection at a specific depth as all the tires in a gear assembly and that the ESWL varies with depth. Figure 1 illustrates the ESWL concept. The equivalent single-wheel load factor (ESWLF) is the ESWL expressed as a fraction of the gear load. The ESWLF would be approximately equal to the fraction of one tire load to the gear load (tire load/gear load) at the surface, and approaches asymptotically an ESWLF value of 1.0 with depth.

2. This report describes the implementation of a computer program to calculate the ESWLF of any gear configuration using Boussinesq solution and vertical deflection as the equivalency parameter. It is not intended to discuss the pros and cons of the concept assumptions, but rather to present and

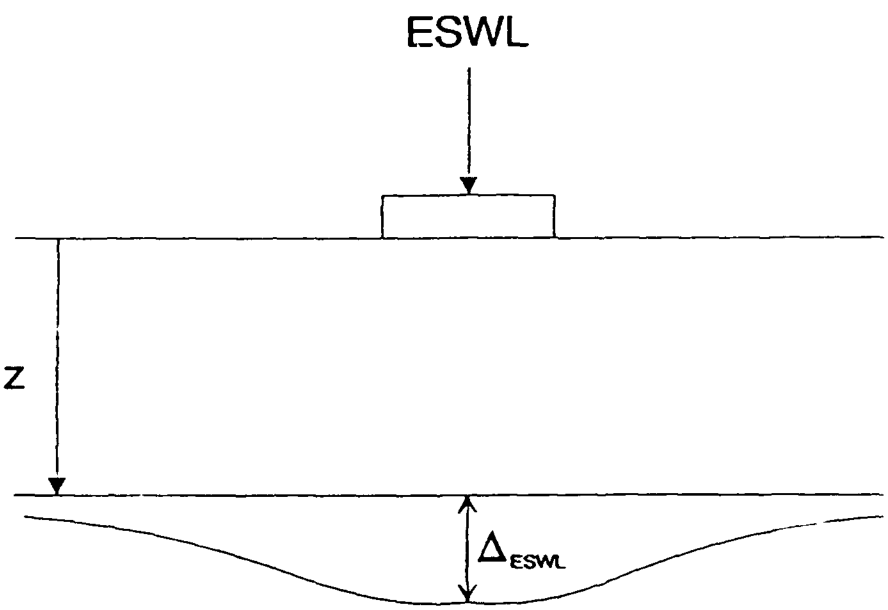
---

* E. J. Yoder and M. W. Witzak. 1975. "Principles of Pavement Design," Wiley-Interscience, NY.

** Boeing Company, Inc. 1963. "A Computer Program for Determining the Flotation Capability of Aircraft Landing Gear Using the CBR Method," Document No. D64088TN, Renton, WA.



a. Deflection under multiple gear



b. Deflection under equivalent single wheel

Figure 1. ESWL analysis for equal deflection criteria

describe the computer program developed. It has been found that better solutions can be achieved if more sophisticated layered elastic system computer programs are used, particularly when a large number of wheels are needed to support heavy aircraft or vehicles. However, for simple gear configurations with few wheels (i.e. twin or twin-tandem gears), the ESWLF program produces reasonable results and is still in common use even for more than four wheels.

PART II: PROGRAMMING PHILOSOPHY AND SYSTEM REQUIREMENTS

3. Emphasis on ease of use, data entry flexibility, and simplicity were major concerns while developing this computer program. Menus, form-like data entry, and windows enhance and simplify the manipulation, calculation, and presentation of the results. The computer code (Appendix A) was written with the structure programming concept in mind to improve code maintainability. The program was compiled using the Microsoft FORTRAN 5.0 compiler and was designed to operate on an IBM Personal Computer or compatible machine under the DOS 3.0 operating system and later versions.

4. The graphics libraries and extensions to the standard FORTRAN included with this compiler were implemented in this program. Although not required, an IBM AT class computer or better with a math coprocessor and 640 KBytes of main memory is recommended.

PART III: USER INSTRUCTIONS

5. The commands and features available in this program are described. An example problem is used to work the user through the steps required to calculate the ESWL. Several conventions are used:

   a. A word or phrase inside brackets means that the user needs to type the word or phrase at the keyboard or press the indicated key. For example: [ENTER] is telling the user to press the enter key, [UP ARROW] means pressing the up-arrow key on the keyboard, and so on.

   b. The example assumes the program is being run from a hard disk with a logical drive C. If the user is running from a hard disk other than C or from a floppy drive, just replace C in the instructions with the corresponding drive letter. For the purpose of this example the drive letter is identified as C:>. This represents the DOS prompt and does not need to be typed.

   c. The program supports the use of the keyboard arrow keys. In the menus, the options can be selected either by moving a highlighted bar to the option desired or pressing the number in brackets preceding the option.

Running the Program

6. Before running the program, the hard disk or floppy disk where it is to be run should be checked to be sure sufficient space is available to hold the input and output data files. To run the program just type C:> ESWLF [ENTER] at the DOS prompt.

7. A welcome screen appears next displaying the program title, author, and date released. After pressing any key, a menu containing the program features or options is displayed (Figure 2). From this menu the entry of data, file manipulation, and viewing of the results can be accomplished. The menu shown in Figure 2 allows the user to build a new data file, load an existing data file, save currently calculated results to disk, view an existing output data file, and exit the program.

Entering Data

8. In general, the steps required to calculate the ESWL consist of:

   a. Gathering the vehicle tire coordinates, tire pressure, and radius of tire contact area.

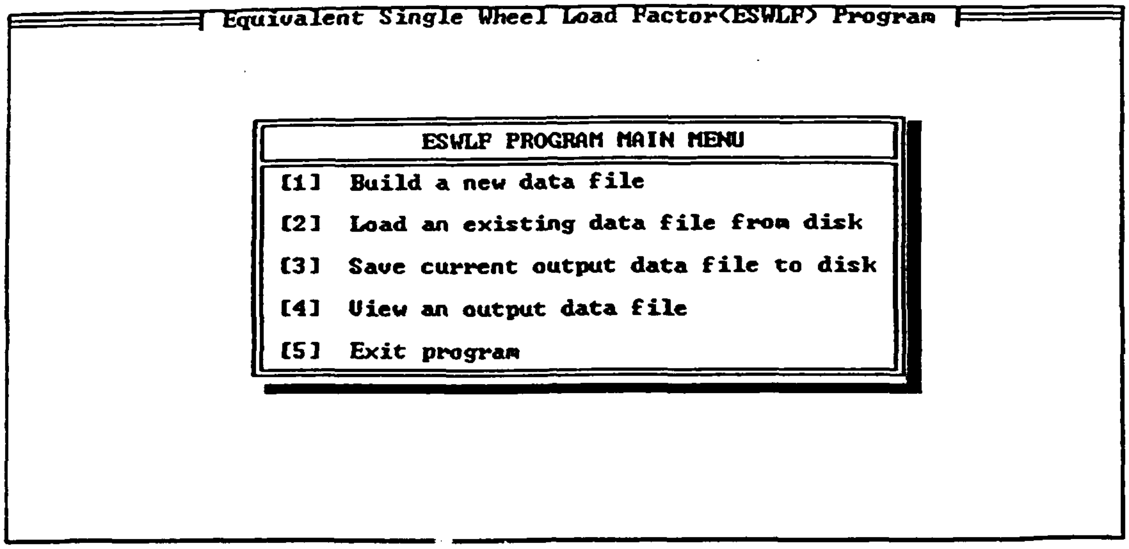


Figure 2. ESWLF program's main menu

b. Building the input data file.

c. Selecting the option to proceed with the calculations.

d. Viewing and/or printing the results obtained.

9. The first step in calculating the ESWL is to build the input data file. To do so the user would select option number one (1) and press [ENTER]. The program will then open a window asking for an input data file name. For simplicity, the file name extensions for the input data file and output data file have been preset to *.IN and *.OUT, respectively. At this point the desired file name, up to eight characters long, can be typed (Figure 3). If the file name already exists, the program will warn the user of such situation and will allow the user to re-enter file name. Pressing the [ESC] key cancels the input and returns the user to the menu.

10. Once the input file name is entered a data entry screen with the required parameters that are needed to perform the calculations is displayed (Figure 4). A description of the parameters in this screen follows:

a. Same contact pressure for all tires? (Y/N)

This item permits the analysis of tires with different contact pressures. However, it is common for the tires in a gear assembly to be of the same size and have the same inflation pressure. Also, in pavement analysis the tire inflation pressure is often assumed to be equal to the tire contact pressure when the tire contact pressure is unknown.

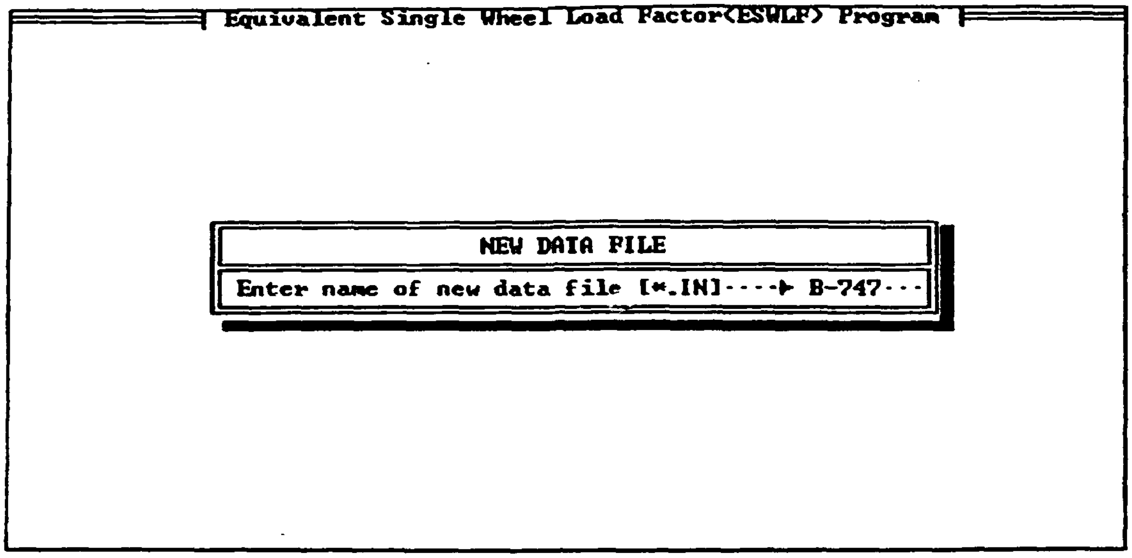


Figure 3. Screen for establishing a new input data file name

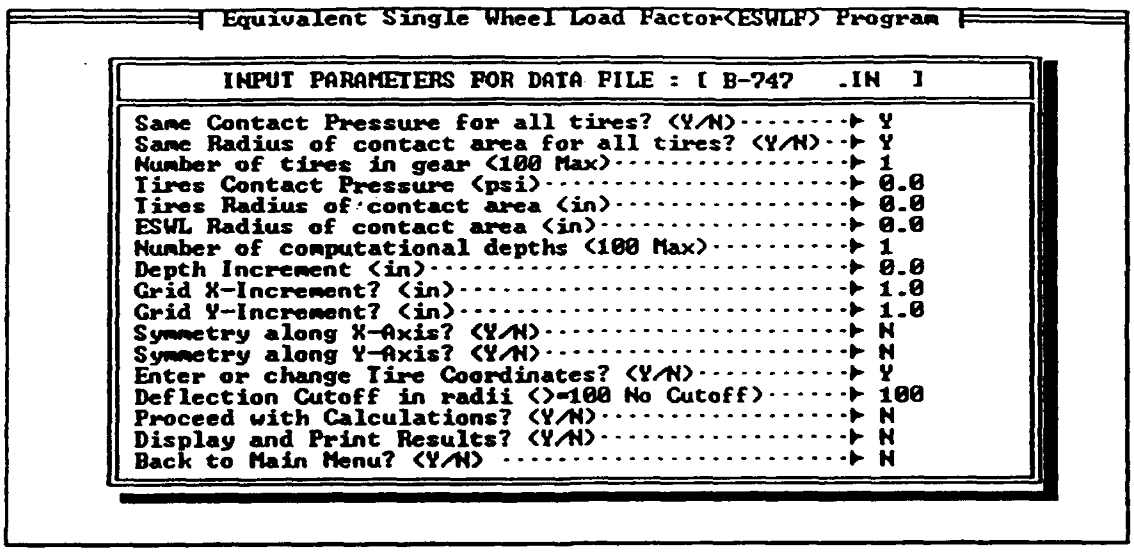


Figure 4. Data entry screen for the ESWLF program

b. Same radius of contact area for all tires? (Y/N)

This item permits the analysis of tires with different radius of contact area. Like the case of the tire pressures, in most situations the radius of contact area in a gear assembly is also considered to be equal for all tires.

c. Number of tires in gear (100 Max)

This is the number of tires to be analyzed. The program allows up to 100 tires.

d. Tire contact pressure (psi)*

If it is decided to have all tires with the same contact pressure, then this item will be active and the value entered will correspond to the tire pressure of all tires. The tire pressure can be computed by dividing the tire load by the contact area or it may be assumed to be equal to the tire inflation pressure.

e. Tire radius of contact area (in.)

If it is decided to have all tires with the same radius of contact area, then this item will be active and the value entered will correspond to the radius of all the tires. Since the Boussinesq theory only considers uniform loads on circular areas, the radius of contact area can be calculated by assuming the contact area to be circular and taking the equation for the area of a circle and solving it for the radius.

f. ESWL radius of contact area (in.)

For the CE procedure this value is always set equal to the radius of one of the tires in the landing gear.

g. Number of computational depths (100 max)

Because the vertical deflection caused by a tire load varies with depth, the program allows the user to compute the ESWLF for up to 100 depths.

h. Depth Increment (in.)

The program uses the number of depths from the previous item and the depth increment to establish the depths at which the ESWLF are to be computed.

i. Grid x-increment (in.) and grid y-increment (in.)

Since this program was designed to handle multiwheel gear configurations, the maximum deflection of such a gear does not necessarily occur under one of the tires. Consequently, several points around the gear must be checked to obtain the maximum value. The program does so by building a grid based on the grid x and y increments entered by the user. The finer the grid generated, the more accurate the results will be. How

---

* A table of factors for converting non-SI units of measurement to SI (metric) units is presented on page 3.

ever, using fine grids also increases the time of the calculations. The origin of the grid, which is independent of the X-Y coordinate system, is automatically set to the tire coordinate of the leftbottom corner of the gear assembly. Figure 5 illustrates the grid system the program would generate for the gear configuration shown.

j. Symmetry along X and Y axes (Y/N)

This feature takes advantage of gear configurations with symmetry along their axes. For example, a gear is symmetric along the X-axis if the tires on the positive Y-axis side and the negative Y-axis side are mirror images of each other. The same analogy applies to the Y-axis. This is important because symmetry greatly reduces the calculation time. Figure 5 illustrates the grid systems the program would generate if symmetry is not assumed along the X and Y axes (Figure 5a) and if symmetry is assumed along both the X and Y axes (Figure 5b). Although the location of the axes is fixed at the center of the gear, it is allowed to set its location at any point.

k. Enter or change tire coordinates? (Y/N)

This entry allows the user to enter or change tire coordinates. When answered affirmative, a window like the one shown in Figure 6 will appear to ask for these values for each of the tires. The tire pressures and radius of contact area are entered only if these values change from tire to tire.

l. Deflection cutoff in radii (>=100 no cutoff)

This is a new parameter not used in the current CE procedure, and is included for research purposes only. If the deflection profile under a wheel load is plotted, it shows that the theoretical deflection only gets to zero at a very long distance from the center of the tire. In reality, this deflection profile reaches zero at considerable shorter offset distances. Figure 7 graphically illustrates this concept. The default value of 100 is used for routine design conditions.

m. Proceed with calculations? (Y/N)

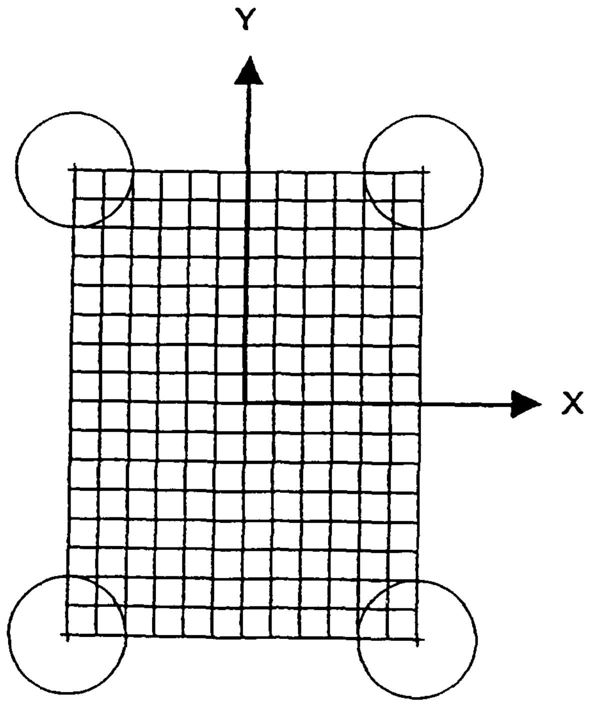
Answer yes to this item if all data have been entered. A window showing the progress of the calculations is displayed. All input parameters must be entered for this option to be active. By default, the program will always create an output data file names ESWLF.OUT. The results in this file may be saved with another name using one of the options from the main menu.

n. Display and Print Results? (Y/N)

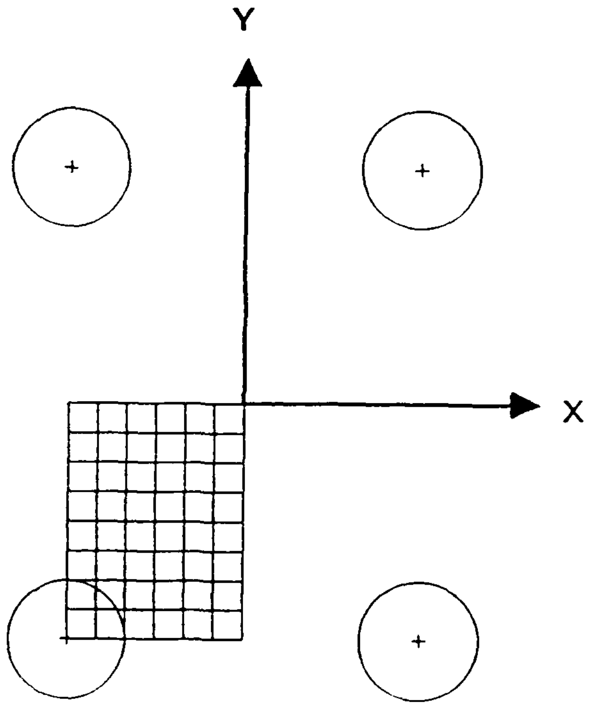
After the calculations have been performed, the user can display or print the results. If no calculations have been performed, a window opens to warn the user to perform the calculations first.

o. Back to main menu? (Y/N)

This option takes the user back to the main menu.



a. No X, Y symmetry



b. With X, Y symmetry

Figure 5. Example grids system generated by the program for a twin-tandem gear

```
INPUT PARAMETERS FOR DATA FILE : [ EXAMPLE .IN  ]

Same Contact Pressure for all tires? (Y/N)········▶ N
Same Radius of contact area for all tires? (Y/N)··▶ N
Number of tires in gear (100 Max)·················▶ 4
Tires C
Tires R      COORDINATES OF TIRE NUMBER   1
ESWL Ra
Number   X-Coord. (in)····················▶ 44.0
Depth I  Y-Coord. (in)····················▶ 0.0
Grid X-  Contact Pressure (psi)···········▶ 200
Grid Y-  Radius of contact area (sq in)···▶ 9.52
Symmetr       Ok ? (Y/N)·················▶  Y
Symmetr
Enter or
Deflection Cutoff in radii (>=100 No Cutoff)······▶ 100
Proceed with Calculations? (Y/N)··················▶ N
Display and Print Results? (Y/N)··················▶ N
Back to Main Menu? (Y/N) ·························▶ N
```

Figure 6. Screen for the input of tire coordinates, tire pressure, and radius of contact area

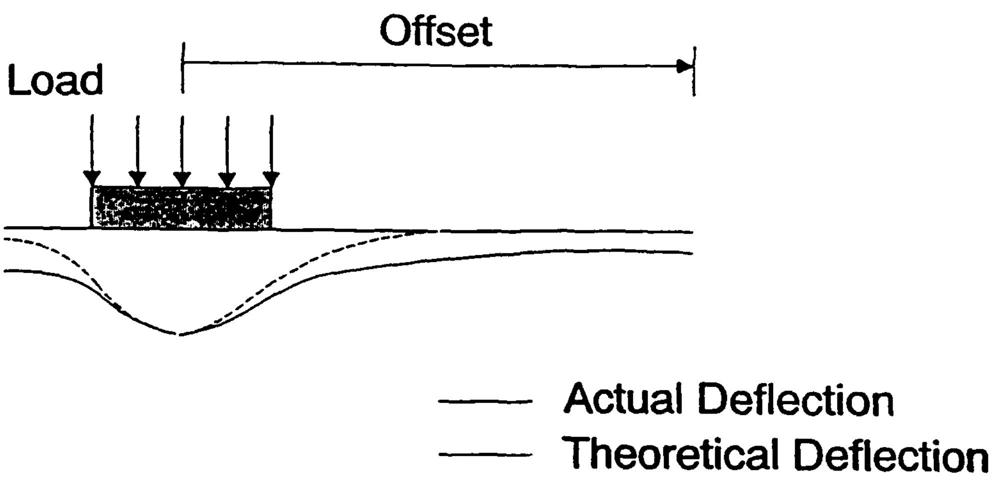


Figure 7. Theoretical and actual deflection profile under a circular load

Example Problem

11. Figures 8 and 9 present the example problem. Figure 10 shows the screen with the input parameters for the example. The arrow keys, page-up, and page-down keys are used to move between the windows input fields to enter the corresponding values. Once all parameters are entered and the calculations performed, the option to display or print the results can be selected. Figure 11 lists the results in a tabular form. This table of results presents the calculated ESWL factors and the x and y location where the maximum deflection occurs for a specific depth. The commands available are displayed at the bottom of the screen. The user may scroll up or down through the table if necessary, print the results, or get back to the data entry screen.

Additional Program Features

12. The program's main menu contains additional features to save the current output data file with another name, view previously computed results, and load previously created input data files.

13. Option [2] of the menu allows the user to retrieve an input data file previously created. When this option is chosen, a list of the available data files is presented from which the user may select one. If no files exist, the program warns the user that no files exist.

14. Option [3] allows the user to save the ESWLF.OUT data file created from a previous calculation. A window then asks for a new name for the output data file. Every time calculations are performed, the program automatically saves the results to the ESWLF.OUT file. Because this operation overwrites the existing ESWLF.OUT file, this option allows the user to save results from previous calculations to a file with another name.

15. Option [4] allows the user to view any output data files previously created and saved with option [3]. A list of available output data files is shown on the screen from which the user can select a file to view.

Example problem:

Compute the equivalent single-wheel load factors (ESWLF) for one side of C-5A main landing gear. Calculate the ESWL down to 100 in. with increments of 10 in. Assume no deflection cutoff distance.

Tire Load = 30,000 lb

Tire Contact Area = 285 sq. in. (measured)

Tire Radius = SQRT (Tire Contact Area/3.1415) = 9.52 in.

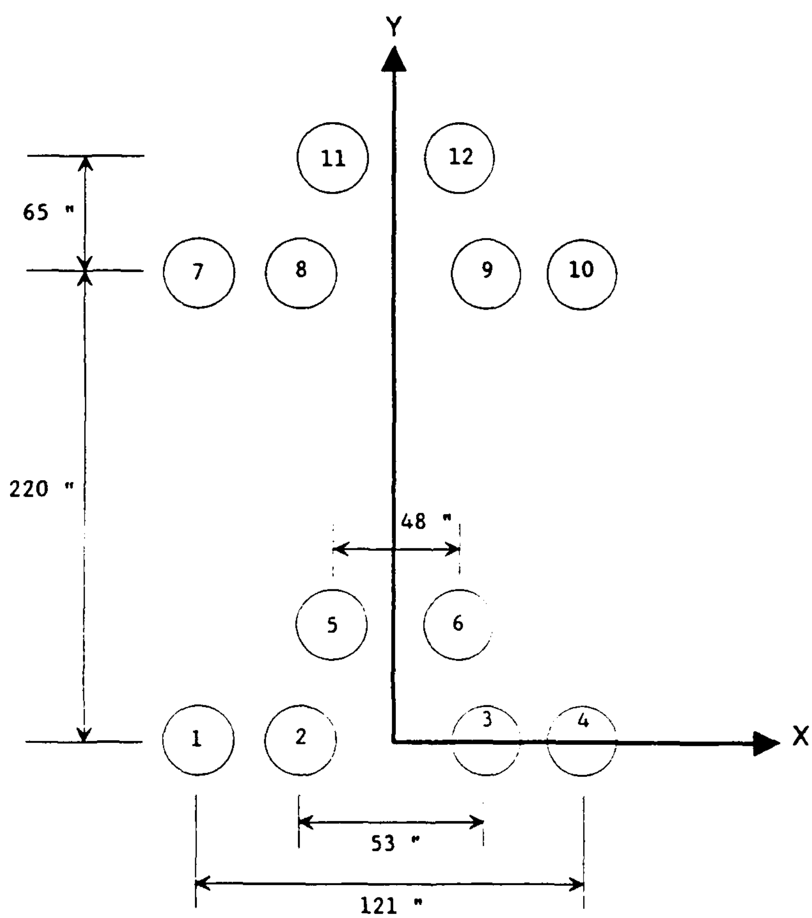


Figure 8. Data for example problem

Solution:

(1) Tire Coordinates

Tire No.	X (in)	Y (in)
1	-60.5	0.0
2	-26.5	0.0
3	26.5	0.0
4	60.5	0.0
5	-24.0	65.0
6	24.0	65.0
7	-60.5	220.0
8	-26.5	220.0
9	26.5	220.0
10	60.5	220.0
11	-24.0	285.0
12	24.0	285.0

(2) Number of depths = 11 (10 increments plus surface deflection)

(3) Depth increment = 10 in.

(4) Symmetry along Y-Axis

(5) No Symmetry along X-Axis

(6) X and Y grid increments = 2 in.

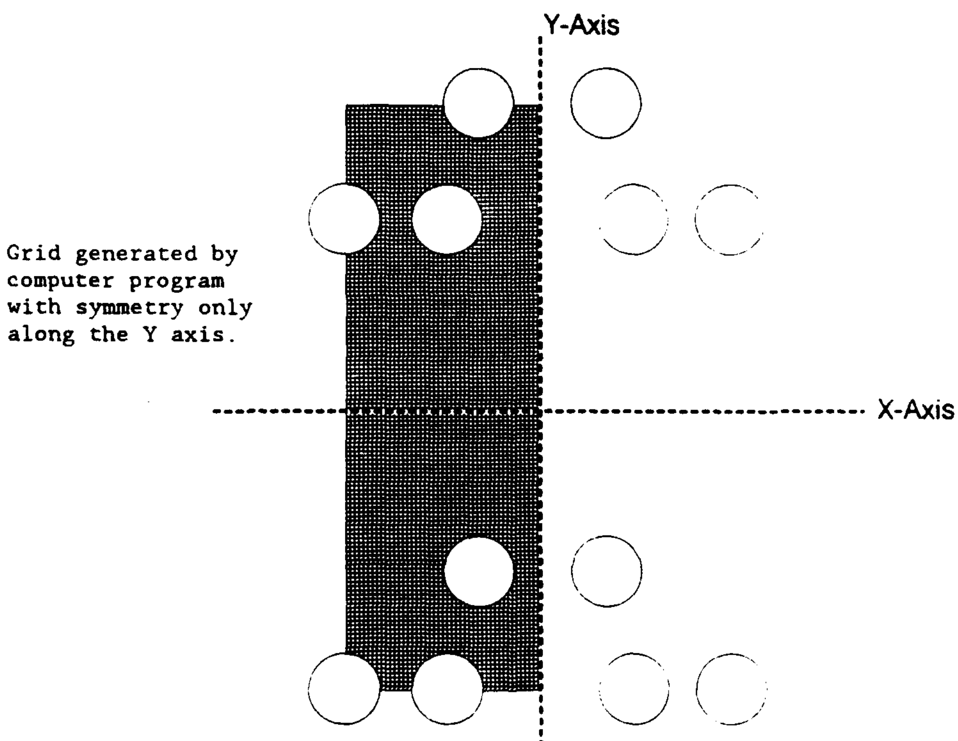


Figure 9. Tire coordinates and grid information for example problem

INPUT PARAMETERS FOR DATA FILE : [C5A-GEAR.IN]	
Same Contact Pressure for all tires? (Y/N)	Y
Same Radius of contact area for all tires? (Y/N)	Y
Number of tires in gear (100 Max)	12
Tire Contact Pressure (psi)	285.00
Tire Radius of contact area (in)	9.52
ESWL Radius of contact area (in)	9.52
Number of computational depths (100 Max)	11
Depth Increment (in)	10.00
Grid X-Increment? (in)	2.00
Grid Y-Increment? (in)	2.00
Symmetry along X-Axis? (Y/N)	N
Symmetry along Y-Axis? (Y/N)	Y
Enter or change Tire Coordinates? (Y/N)	N
Deflection Cutoff in radii (>=100 No Cutoff)	100.00
Proceed with Calculations? (Y/N)	Y
Display and Print Results? (Y/N)	N
Back to Main Menu? (Y/N)	N

a. Input parameters for example problem

Tire #	X-coord(in)	Y-coord(in)	Radius(in)	Press(psi)
1	-60.50	.00	9.52	285.00
2	-26.50	.00	9.52	285.00
3	26.50	.00	9.52	285.00
4	60.50	.00	9.52	285.00
5	-24.00	65.00	9.52	285.00
6	24.00	65.00	9.52	285.00
7	-60.50	220.00	9.52	285.00
8	-26.50	220.00	9.52	285.00
9	26.50	220.00	9.52	285.00
10	60.50	220.00	9.52	285.00
11	-24.00	285.00	9.52	285.00
12	24.00	285.00	9.52	285.00

b. C-5A landing gear tire coordinates

Figure 10. Example problem computer screens

DEPTH(in)	ESWLF	XMAX(in)	YMAX(in)
.00	.091	-26.50	220.00
10.00	.152	-26.50	220.00
20.00	.197	-28.50	220.00
30.00	.247	-28.50	222.00
40.00	.296	-24.50	224.00
50.00	.344	-12.50	226.00
60.00	.392	-.50	230.00
70.00	.433	-.50	230.00
80.00	.470	-.50	228.00
90.00	.503	-.50	228.00
100.00	.533	-.50	224.00

Figure 11. Table showing the results obtained from the example problem

PART IV: CONCLUSIONS AND RECOMMENDATIONS

Conclusions

16. A computer program for the calculation of the equivalent single-wheel load based on the Boussinesq solution and using deflection as the equivalency parameter was presented. The following conclusions were made:

a. Implementing the ESWL concept into a user friendly computer program is a tedious task and, although the concept is fairly simple, the numerical analysis for its solution can become very complex.

b. The computer program resulting from the implementation of ESWLF concept into a FORTRAN code produced an efficient and easy to use pavement analysis tool. Sufficiently accurate results for pavement analysis can be obtained with minimum interaction between the user and the program, thus hiding the user from the complex mathematical algorithms involved in the solution.

c. The easy to use interface greatly enhances the program aesthetics and functionality, and eliminates cumbersome and confusing data entry screens.

Recommendations

17. Based on the experience gained while developing and using this program, it is recommended that:

a. Grid intervals of about half the contact area radius give sufficiently accurate results.

b. Gear symmetry should be used whenever possible to reduce calculations.

APPENDIX A: COMPUTER PROGRAM SOURCE CODE

```
$LARGE
C*********************************************************************
C   Equivalent Single Wheel Load Factor program
C   written by:  Carlos R. Gonzalez
C                Pavement System Division
C                Geotechnical Laboratory
C                Waterways Experiment Station
C                Vicksburg, Mississippi
C*********************************************************************

      INCLUDE 'FGRAPH.FI'

      PROGRAM ESWLFACTOR

      CHARACTER*12 FNAME, OUTFILE, WORKER, FSPEC*40
      CHARACTER*3 EXT
      LOGICAL QUIT

      CALL C_FILLSTR(FNAME,' ')
      CALL C_FILLSTR(WORKER,' ')
      CALL C_FILLSTR(OUTFILE,' ')
      CALL C_NUMOFF
      CALL Hello
      CALL ScreenBk
      QUIT = .FALSE.
      DO WHILE(.NOT.QUIT)
        CALL MainMenu(iPick)
        SELECT CASE(iPick)
         CASE(1)
           EXT = 'IN'
           CALL NewFNAME(WORKER,EXT)
           IF (WORKER.NE.' ') THEN
             FNAME = WORKER
             CALL ESWLF(FNAME)
             CALL C_FILLSTR(WORKER,' ')
           END IF
         CASE(2)
           FSPEC = '*.IN'
           CALL GetFile(WORKER,FSPEC)
           IF (WORKER.NE.' ') THEN
             FNAME = WORKER
             CALL C_FILLSTR(WORKER,' ')
           END IF
           IF (FNAME.NE.' ') THEN
             CALL ESWLF(FNAME)
             CALL C_FILLSTR(WORKER,' ')
           END IF
         CASE(3)
           EXT = 'OUT'
           CALL NewFNAME(WORKER,EXT)
           IF (WORKER.NE.' ') THEN
             CALL SRESULTS(WORKER)
             CALL C_FILLSTR(WORKER,' ')
           END IF
         CASE(4)
```

```
          FSPEC = '*.OUT'
          CALL GetFile(WORKER,FSPEC)
          IF (WORKER.NE.' ') THEN
            OUTFILE = WORKER
            CALL C_FILLSTR(WORKER,' ')
          END IF
          IF (OUTFILE.NE.' ') THEN
            CALL ShowRes(WORKER,OUTFILE)
            CALL C_FILLSTR(WORKER,' ')
            CALL C_FILLSTR(OUTFILE,' ')
          END IF
        CASE(5)
          CALL ShutDown
          QUIT = .TRUE.
        CASE DEFAULT
      END SELECT
      END DO

      STOP ' '
      END
C***********************************************************************
      SUBROUTINE MainMenu(iPick)

      CHARACTER*78 MA(9), Title

      Nitems = 9
      Title = ' ESWLF PROGRAM MAIN MENU'
      MA = ' '
      MA(1) = ' [1]  Build a new data file'
      MA(3) = ' [2]  Load an existing data file from disk'
      MA(5) = ' [3]  Save current output data file to disk'
      MA(7) = ' [4]  View an output data file'
      MA(9) = ' [5]  Exit program'

      CALL C_CMENU(Nitems,MA,Title,iPick)

      RETURN
      END
C***********************************************************************
      SUBROUTINE NewFNAME(FNAME,EXT)

      CHARACTER*78 BA(1), Title, MSG
      CHARACTER worker*8, FNAME*12, EXT*3
      LOGICAL Escape, exists, VALID
      INTEGER row, col

 99   Nitems = 1
      IF (EXT.EQ.'OUT') THEN
        Title = 'OUTPUT DATA FILE'
        BA(1) = ' Enter name of output data file [*.OUT]úúúú■'
      ELSE
        Title = 'NEW DATA FILE'
        BA(1) = ' Enter name of new data file [*.IN]úúúú■'
      END IF
```

```
      worker = ' '
      CALL ScreenBk
      CALL C_CBOX(Nitems,BA,Title,row,col,8)
 77   CALL C_GETSTR(row,col,worker,8,.TRUE.,.FALSE.,.FALSE.,
     &              RR,II,IERR,Escape,IARROW)
      IF (Escape) THEN
        FNAME = ' '
      ELSE
        IF (worker.EQ.' ') GOTO 77
        CALL C_CHKFNAME(worker,VALID)
        IF (.NOT.VALID) THEN
          worker = ' '
          GOTO 77
        END IF
        FNAME = worker//'.'//EXT
        INQUIRE(FILE=FNAME,EXIST=exists)
        IF (exists) THEN
          TITLE = 'FILE ALREADY EXISTS!'
          MSG = ' Press any key and enter a new file name.'
          CALL ERRMSG(TITLE,MSG)
          CALL ScreenBk
          FNAME = ' '
          GOTO 99
        END IF
      END IF
      CALL ScreenBk

      RETURN
      END
C**************************************************************************
      SUBROUTINE GetFile(FNAME,worker)

      CHARACTER FNAME*12, FSPEC*40, worker*12
      CHARACTER*78 TITLE, MSG
c      LOGICAL ERROR

      FSPEC = worker
      CALL ScreenBk
      CALL C_DIRFILES(FSPEC,NFILES,FNAME)
      CALL ScreenBk
      CALL C_CURSOROFF
c      IF (ERROR) THEN
c        TITLE = 'DISK OR MEMORY PROBLEMS'
c        MSG = ' CANNOT READ DIRECTORY!'
c        CALL ERRMSG(TITLE,MSG)
c        CALL ScreenBk
c        FNAME = ' '
c      ELSE
       IF (NFILES.GT.0) THEN
         ! Ok to return FNAME
       ELSE
         TITLE = 'FILES NOT FOUND'
         MSG = ' NO FILES FOUND MATCHING SPEC: '//FSPEC
         CALL ERRMSG(TITLE,MSG)
         CALL ScreenBk
```

```
          FNAME = ' '
        END IF
c      END IF
      CALL C_CURSORON

      RETURN
      END
C***********************************************************************
      SUBROUTINE SRESULTS(FNAME)
      CHARACTER*12 FNAME
      CHARACTER*80 CDUMMY
      LOGICAL exists

      IDISK1 = 3
      IDISK2 = 4
      INQUIRE(FILE='ESWLF.OUT',EXIST=exists)
      IF (.NOT. exists) THEN
        CALL NoResults
        RETURN
      ELSE
        OPEN(IDISK1,FILE='ESWLF.OUT',STATUS='OLD')
        OPEN(IDISK2,FILE=FNAME,STATUS='NEW')
         DO i=1, 1000
           READ(IDISK1,'(A)',END=10) CDUMMY
           WRITE(IDISK2,'(A)') CDUMMY
         END DO
 10     CLOSE(IDISK1)
        CLOSE(IDISK2)
      END IF

      RETURN
      END
C***********************************************************************
      SUBROUTINE ERRMSG(TITLE,MSG)
      CHARACTER*78 MSG, BA(1), Title

      Nitems = 1
      BA(1) = MSG
      CALL C_CURSOROFF
      CALL C_CBOXFLASH(Nitems,BA,Title)
      CALL C_WAITFORKEY
      CALL C_CURSORON

      RETURN
      END
C*********************************************************************
      SUBROUTINE Hello

      CHARACTER BA(11)*78  !BoxArray
      CHARACTER Title*78

      BA = ' '
      NItems = 11
      Title = ' EQUIVALENT SINGLE WHEEL LOAD FACTOR PROGRAM'
      BA(2) = '                    written by'
```

```
      BA(4) = '              Carlos R. Gonzalez'
      BA(5) = '            Pavement System Division'
      BA(6) = '            Geotechnical  Laboratory'
      BA(7) = ' US Army Engineer Waterways Experiment Station'
      BA(8) = '            Vicksburg,  Mississippi'
      BA(9) = '                 July 1991'
      BA(11)= '           Press any key to continue.'

      CALL C_CURSOROFF
      CALL C_CLRSCR
      CALL C_CBOX(Nitems,BA,Title,ir,ic,0)
      CALL C_WAITFORKEY
      CALL C_CURSORON

      RETURN
      END
C*********************************************************************
      SUBROUTINE Wait(MSG)

      CHARACTER BA(1)*78  !BoxArray
      CHARACTER Title*78
      CHARACTER MSG

      BA = ' '
      NItems = 1
      Title = MSG
      BA(1) = ' Please wait...'
      CALL C_CURSOROFF
      CALL C_CBOX(Nitems,BA,Title,ir,ic,0)
      CALL C_CURSORON
      CALL ScreenBk

      RETURN
      END
C*********************************************************************
      SUBROUTINE ScreenBk

      CHARACTER tp*80, bt*80

      tp = 'Equivalent Single Wheel Load Factor(ESWLF) Program'
      bt = 'US Army Engineer Waterways Experiment Station'
      CALL C_SCREENBK(tp,bt,.FALSE.)

      RETURN
      END
C*********************************************************************
      SUBROUTINE ShutDown

      CALL C_CLRSCR

      RETURN
      END
C*********************************************************************
      SUBROUTINE StatusBox
```

```
      CHARACTER BA(3)*78  !BoxArray
      CHARACTER Title*78

      BA = ' '
      NItems = 3
      Title = ' PERFORMING CALCULATIONS'

      CALL C_CURSOROFF
      CALL C_CBOXFLASH(Nitems,BA,Title)
      CALL C_CURSORON

      RETURN
      END
C***********************************************************************
      SUBROUTINE STATUS(ICOUNT, ITOTAL)

      INCLUDE 'FGRAPH.FD'

      RECORD /rccoord/ s
      INTEGER*2 r,c
      INTEGER*2 dummy2,fg
      INTEGER*4 dummy4,bk
      CHARACTER pct*4, strng*80

      fg = GetTextColor()
      bk = GetBkColor()
      dummy2 = SetTextColor(15)   !white
      dummy4 = SetBkColor(4)      !blue
      X = ICOUNT
      XT = ITOTAL
      xpct = (X/XT)*100.
      ipct = xpct
      WRITE(pct,'(I3)') ipct
      strng = pct//'% COMPLETE'
      r = 13
      c = (80-LEN_TRIM(strng))/2 + 1
      CALL SetTextPosition(r,c,s)
      CALL OutText(strng(1:LEN_TRIM(strng)))
      dummy2 = SetTextColor(fg)
      dummy4 = SetBkColor(bk)

      RETURN
      END
C***********************************************************************
      SUBROUTINE ESWLF(FNAME)

      INCLUDE 'FGRAPH.FD'

      CHARACTER FNAME*12, OUTFILE*12
      LOGICAL QUIT, CHANGES
      INTEGER   c
      INTEGER   jr, jc
      INTEGER   row(20), col
      LOGICAL SAMEp, SAMEr, SAVE
      DIMENSION X(100), Y(100), PRESS(100), RADIUS(100), Z(100)
```

```
      CHARACTER*10 Field(20)
      DIMENSION ICODE(20)

      CHARACTER*20 Ch
      LOGICAL      Ok

      LOGICAL Fexists
      CHARACTER INTF*10

      LOGICAL Escape
      CHARACTER YESNO*1, FNUM3*3, FNUM7*7
      INTEGER*2 ITEM

C      CALL Wait('LOADING PROGRAM')
      CALL ReadFile(FNAME,NTIRES,X,Y,PRESS,RADIUS,RESW,NDEPTHS,
     &              DZ,GDX,GDY,IXSYM,IYSYM,RCUT,Fexists)

      SAMEp = .TRUE.
      SAMEr = .TRUE.

      NItems = 17

      IF (.NOT.Fexists) THEN
        DO i=1, NItems
          DO j=1, 10
            Field(i)(j:j) = ' '
          END DO
        END DO
        DO i=1, 14
          ICODE(i) = 0
        END DO
        NTIRES = 1         ! set some defaults
        NDEPTHS = 1
        PRESS(1) = 0.0
        RADIUS(1) = 0.0
        RESW = 0.0
        DZ = 0.0
        GDX = 1.0
        GDY = 1.0
        RCUT = 100.0
        Field(1)  = 'Y'
        Field(2)  = 'Y'
        Field(3)  = '1'
        Field(4)  = '0.0'
        Field(5)  = '0.0'
        Field(6)  = '0.0'
        Field(7)  = '1'
        Field(8)  = '0.0'
        Field(9)  = '1.0'
        Field(10) = '1.0'
        Field(11) = 'N'
        Field(12) = 'N'
        Field(13) = 'Y'
        Field(14) = '100'
        Field(15) = 'N'
```

```
     Field(16) = 'N'
     Field(17) = 'N'
     ICODE(1)  = 1
     ICODE(2)  = 1
     ICODE(3)  = 1
     ICODE(7)  = 1
     ICODE(8)  = 1
     ICODE(9)  = 1
     ICODE(10) = 1
     ICODE(11) = 1
     ICODE(12) = 1
     ICODE(13) = 1
     ICODE(14) = 1
     CHANGES = .TRUE.
   ELSE
     DO i=1, NTIRES
       IF ( PRESS(1).NE.PRESS(i) ) SAMEp = .FALSE.
       IF ( RADIUS(1).NE.RADIUS(i) ) SAMEp = .FALSE.
     END DO
     DO i=1, 14
       ICODE(i) = 1
     END DO
     CHANGES = .FALSE.
     DO i=1, NItems
       IF (SAMEp) THEN
         Field(1) = 'Y'
       ELSE
         Field(1) = 'N'
       END IF
       IF (SAMEr) THEN
         Field(2) = 'Y'
       ELSE
         Field(2) = 'N'
       END IF
       WRITE(INTF,'(I10)') NTIRES
       READ(INTF,'(A10)') Field(3)
       IF (Field(1).EQ.'Y') THEN
         WRITE(INTF,'(F10.2)') PRESS(1)
         READ(INTF,'(A10)') Field(4)
       ELSE
         Field(4) = '----------'
       END IF
       IF (Field(2).EQ.'Y') THEN
         WRITE(INTF,'(F10.2)') RADIUS(1)
         READ(INTF,'(A10)') Field(5)
       ELSE
         Field(5) = '----------'
       END IF
       WRITE(INTF,'(F10.2)') RESW
       READ(INTF,'(A10)') Field(6)
       WRITE(INTF,'(I10)') NDEPTHS
       READ(INTF,'(A10)') Field(7)
       WRITE(INTF,'(F10.2)') DZ
       READ(INTF,'(A10)') Field(8)
       IF (DZ.EQ.0.0) Field(8) = '0.0'
```

```
          WRITE(INTF,'(F10.2)') GDX
          READ(INTF,'(A10)') Field(9)
          WRITE(INTF,'(F10.2)') GDY
          READ(INTF,'(A10)') Field(10)
          IF (IXSYM.EQ.1) THEN
            Field(11) = 'Y'
          ELSE
            Field(11) = 'N'
          END IF
          IF (IYSYM.EQ.1) THEN
            Field(12) = 'Y'
          ELSE
            Field(12) = 'N'
          END IF
          Field(13) = 'N'
          WRITE(INTF,'(F10.2)') RCUT
          READ(INTF,'(A10)') Field(14)
          Field(15) = 'N'
          Field(16) = 'N'
          Field(17) = 'N'
        END DO
      END IF
      DO i=1, 17
        CALL C_JUSTIFY(Field(i),.FALSE.)
      END DO

      CALL DrawForm(jr,jc,FNAME)
      c = jc+1
      row(1) = jr
      DO i=2, NItems
        row(i) = row(i-1) + 1
      END DO

      DO i=1, NItems
        CALL WriteField(row(i),c,Field(i))
      END DO

      ITEM = 1
      QUIT = .FALSE.
      DO WHILE (.NOT.QUIT)

      SELECT CASE(ITEM)

      CASE(1)
99    c = jc+1
      col = jc
      OK = .FALSE.
      DO WHILE (.NOT. OK)
        YESNO = Field(1)(1:1)
        ch(1:1) = YESNO
        CALL C_GETSTR(row(1),col,YESNO,1,.TRUE.,.FALSE.,.FALSE.,
     &             RR,II,IERR,Escape,IARROW)
        SELECT CASE (YESNO(1:1))
         CASE('y','Y')
          SAMEp = .TRUE.
```

```
          OK = .TRUE.
          WRITE(INTF,'(F10.2)') PRESS(1)
          READ(INTF,'(A10)') Field(4)
          CALL C_JUSTIFY(Field(4),.FALSE.)
          CALL WriteField(row(4),c,Field(4))
        CASE('n','N')
          SAMEp = .FALSE.
          OK = .TRUE.
          Field(4) = '----------'
          CALL WriteField(row(4),c,Field(4))
        CASE DEFAULT
      END SELECT
      SELECT CASE(IARROW)
        CASE(1:9)
          SELECT CASE(YESNO)
          CASE('Y','y','N','n')
            OK = .TRUE.
          CASE DEFAULT
            OK = .FALSE.
          END SELECT
        CASE DEFAULT
          OK = .FALSE.
      END SELECT
    END DO
    IF (ch(1:1).NE.YESNO) THEN
      CHANGES = .TRUE.
      ICODE(1) = 1
      Field(1)(1:1) = YESNO
    ELSE
      Field(1)(1:1) = ch(1:1)
    END IF
    CALL C_JUSTIFY(Field(1),.FALSE.)
    CALL WriteField(row(1),c,Field(1))

    CASE(2)
    OK = .FALSE.
    DO WHILE (.NOT. OK)
      YESNO = Field(2)(1:1)
      ch(1:1) = YESNO
      CALL C_GETSTR(row(2),col,YESNO,1,.TRUE.,.FALSE.,.FALSE.,
&                   RR,I1,IERR,Escape,IARROW)
      SELECT CASE (YESNO(1:1))
       CASE('y','Y')
        SAMEr = .TRUE.
        OK = .TRUE.
        WRITE(INTF,'(F10.2)') RADIUS(1)
        READ(INTF,'(A10)') Field(5)
        CALL C_JUSTIFY(Field(5),.FALSE.)
        CALL WriteField(row(5),c,Field(5))
      CASE('n','N')
        SAMEr = .FALSE.
        OK = .TRUE.
        Field(5) = '----------'
        CALL WriteField(row(5),c,Field(5))
      CASE DEFAULT
```

```
      END SELECT
      SELECT CASE(IARROW)
        CASE(1:9)
          SELECT CASE(YESNO)
          CASE('Y','y','N','n')
            OK = .TRUE.
          CASE DEFAULT
            OK = .FALSE.
          END SELECT
        CASE DEFAULT
          OK = .FALSE.
      END SELECT
    END DO
    IF (ch(1:1).NE.YESNO) THEN
      CHANGES = .TRUE.
      ICODE(2) = 1
      Field(2)(1:1) = YESNO
    ELSE
      Field(2)(1:1) = ch(1:1)
    END IF
    CALL C_JUSTIFY(Field(2),.FALSE.)
    CALL WriteField(row(2),c,Field(2))

    CASE(3)
    OK = .FALSE.
    DO WHILE (.NOT. OK)
      FNUM3 = Field(3)(1:3)
      II = NTIRES
      IN = NTIRES
      CALL C_GETSTR(row(3),col,FNUM3,3,.FALSE.,.TRUE.,.FALSE.,
   &              RR,II,IERR,Escape,IARROW)
      IF (IERR.EQ.0) THEN
        IF (II.GE.1 .AND. II.LE.100) THEN
          OK = .TRUE.
          NTIRES = II
          Field(3)(1:10) = FNUM3
          ICODE(3) = 1
        END IF
      END IF
      SELECT CASE(IARROW)
        CASE(1:9)
          OK = .TRUE.
        CASE DEFAULT
          OK = .FALSE.
      END SELECT
    END DO
    CALL WriteField(row(3),c,Field(3))
    IF (IN.NE.NTIRES) THEN
      CHANGES = .TRUE.
      Field(13)(1:1) = 'Y'
    END IF

    CASE(4)
    IF (SAMEp) THEN
      DO i=1,NTIRES
```

```
      PRESS(i) = PRESS(1)
    END DO
    OK = .FALSE.
    DO WHILE (.NOT. OK)
      FNUM7 = Field(4)(1:7)
      RR = PRESS(1)
      RN = RR
      CALL C_GETSTR(row(4),col,FNUM7,7,.FALSE.,.TRUE.,.TRUE.,
&                  RR,II,IERR,Escape,IARROW)
      IF (IERR.EQ.0) THEN
        IF (RR.GE.1.0 .AND. RR.LE.1000.0) THEN
          OK = .TRUE.
          DO i=1, NTIRES
            PRESS(i) = RR
          END DO
          Field(4)(1:10) = FNUM7
          ICODE(4) = 1
        END IF
      END IF
      SELECT CASE(IARROW)
        CASE(1:9)
          OK = .TRUE.
        CASE DEFAULT
          OK = .FALSE.
      END SELECT
    END DO
    CALL WriteField(row(4),c,Field(4))
    IF (RN .NE. PRESS(1)) CHANGES = .TRUE.
  END IF

  CASE(5)
  IF (SAMEr) THEN
    DO i=1,NTIRES
      RADIUS(i) = RADIUS(1)
    END DO
    OK = .FALSE.
    DO WHILE (.NOT. OK)
      FNUM7 = Field(5)(1:7)
      RR = RADIUS(1)
      RN = RR
      CALL C_GETSTR(row(5),col,FNUM7,7,.FALSE.,.TRUE.,.TRUE.,
&                  RR,II,IERR,Escape,IARROW)
      IF (IERR.EQ.0) THEN
        IF (RR.GE.0.1 .AND. RR.LE.1000.0) THEN
          OK = .TRUE.
          DO i=1, NTIRES
            RADIUS(i) = RR
          END DO
          Field(5)(1:10) = FNUM7
          ICODE(5) = 1
        END IF
      END IF
      SELECT CASE(IARROW)
        CASE(1:9)
          OK = .TRUE.
```

```
          CASE DEFAULT
            OK = .FALSE.
        END SELECT
      END DO
    END IF
    CALL WriteField(row(5),c,Field(5))

    CASE(6)
    OK = .FALSE.
    DO WHILE (.NOT. OK)
      IF (ICODE(6).EQ.0) THEN
        FNUM7 = Field(5)(1:7)
        RR = RADIUS(1)
      ELSE
        FNUM7 = Field(6)(1:7)
        RR = RESW
      END IF
      RN = RR
      CALL C_GETSTR(row(6),col,FNUM7,7,.FALSE.,.TRUE.,.TRUE.,
   &                RR,II,IERR,Escape,IARROW)
      IF (IERR.EQ.0) THEN
        IF (RR.GE.0.1 .AND. RR.LE.1000.0) THEN
          OK = .TRUE.
          RESW = RR
          Field(6)(1:10) = FNUM7
          ICODE(6) = 1
        END IF
      END IF
      SELECT CASE(IARROW)
        CASE(1:9)
          OK = .TRUE.
        CASE DEFAULT
          OK = .FALSE.
      END SELECT
    END DO
    CALL WriteField(row(6),c,Field(6))
    IF (RN .NE. RESW) CHANGES = .TRUE.

    CASE(7)
    OK = .FALSE.
    DO WHILE (.NOT. OK)
      FNUM3 = Field(7)(1:3)
      II = NDEPTHS
      IN = II
      CALL C_GETSTR(row(7),col,FNUM3,3,.FALSE.,.TRUE.,.FALSE.,
   &                RR,II,IERR,Escape,IARROW)
      IF (IERR.EQ.0) THEN
        IF (II.GE.1 .AND. II.LE.100) THEN
          OK = .TRUE.
          NDEPTHS = II
          Field(7)(1:10) = FNUM3
          ICODE(7) = 1
        END IF
      END IF
      SELECT CASE(IARROW)
```

```
          CASE(1:9)
            OK = .TRUE.
          CASE DEFAULT
            OK = .FALSE.
        END SELECT
      END DO
      IF (IN .NE. NDEPTHS) CHANGES = .TRUE.
      CALL WriteField(row(7),c,Field(7))

      CASE(8)
      OK = .FALSE.
      DO WHILE (.NOT. OK)
        FNUM7 = Field(8)(1:7)
        RR = DZ
        RN = RR
        CALL C_GETSTR(row(8),col,FNUM7,7,.FALSE.,.TRUE.,.TRUE.,
     &             RR,II,IERR,Escape,IARROW)
        IF (IERR.EQ.0) THEN
          IF (RR.GE.0.0 .AND. RR.LE.1000.0) THEN
            OK = .TRUE.
            DZ = RR
            IF (DZ .EQ. 0.0) THEN
              FNUM7 = '0.0'
              Field(7) = '1'
              NDEPTHS = 1
              ICODE(7) = 1
              CALL WriteField(row(7),c,Field(7))
            END IF
            Field(8)(1:10) = FNUM7
            ICODE(8) = 1
          END IF
        END IF
        SELECT CASE(IARROW)
          CASE(1:9)
            OK = .TRUE.
          CASE DEFAULT
            OK = .FALSE.
        END SELECT
      END DO
      Z(1) = 0.0
      DO i=2, NDEPTHS
        Z(i) = Z(i-1) + DZ
      END DO
      CALL WriteField(row(8),c,Field(8))
      IF (RN .NE. DZ) CHANGES = .TRUE.

      CASE(9)
      OK = .FALSE.
      DO WHILE (.NOT. OK)
        FNUM7 = Field(9)(1:7)
        RR = GDX
        RN = RR
        CALL C_GETSTR(row(9),col,FNUM7,7,.FALSE.,.TRUE.,.TRUE.,
     &             RR,II,IERR,Escape,IARROW)
        IF (IERR.EQ.0) THEN
```

```
      IF (RR.GE.1.0 .AND. RR.LE.1000.0) THEN
        OK = .TRUE.
        GDX = RR
        Field(9)(1:10) = FNUM7
        ICODE(9) = 1
      END IF
    END IF
    SELECT CASE(IARROW)
      CASE(1:9)
        OK = .TRUE.
      CASE DEFAULT
        OK = .FALSE.
    END SELECT
  END DO
  CALL WriteField(row(9),c,Field(9))
  IF (RN .NE. GDX) CHANGES = .TRUE.

  CASE(10)
  OK = .FALSE.
  DO WHILE (.NOT. OK)
    FNUM7 = Field(10)(1:7)
    RR = GDY
    RN = RR
    CALL C_GETSTR(row(10),col,FNUM7,7,.FALSE.,.TRUE.,.TRUE.,
&              RR,II,IERR,Escape,IARROW)
    IF (IERR.EQ.0) THEN
      IF (RR.GE.1.0 .AND. RR.LE.1000.0) THEN
        OK = .TRUE.
        GDY = RR
        Field(10)(1:10) = FNUM7
        ICODE(10) = 1
      END IF
    END IF
    SELECT CASE(IARROW)
      CASE(1:9)
        OK = .TRUE.
      CASE DEFAULT
        OK = .FALSE.
    END SELECT
  END DO
  CALL WriteField(row(10),c,Field(10))
  IF (RN .NE. GDY) CHANGES = .TRUE.

  CASE(11)
  OK = .FALSE.
  DO WHILE (.NOT. OK)
    YESNO = Field(11)(1:1)
    ch(1:1) = YESNO
    CALL C_GETSTR(row(11),col,YESNO,1,.TRUE.,.FALSE.,.FALSE.,
&              RR,II,IERR,Escape,IARROW)
    SELECT CASE (YESNO(1:1))
    CASE('y','Y')
      IXSYM = 1
      OK = .TRUE.
    CASE('n','N')
```

```
      IXSYM = 0
      OK = .TRUE.
    CASE DEFAULT
    END SELECT
    SELECT CASE(IARROW)
      CASE(1:9)
        SELECT CASE(YESNO)
        CASE('Y','y','N','n')
          OK = .TRUE.
        CASE DEFAULT
          OK = .FALSE.
        END SELECT
      CASE DEFAULT
        OK = .FALSE.
    END SELECT
  END DO
  IF (ch(1:1).NE.YESNO) THEN
    CHANGES = .TRUE.
    ICODE(11) = 1
    Field(11) = YESNO
  ELSE
    Field(11) = ch(1:1)
  END IF
  CALL C_JUSTIFY(Field(11),.FALSE.)
  CALL WriteField(row(11),c,Field(11))

  CASE(12)
  OK = .FALSE.
  DO WHILE (.NOT. OK)
    YESNO = Field(12)(1:1)
    ch(1:1) = YESNO
    CALL C_GETSTR(row(12),col,YESNO,1,.TRUE.,.FALSE.,.FALSE.,
&              RR,II,IERR,Escape,IARROW)
    SELECT CASE (YESNO(1:1))
    CASE('y','Y')
      IYSYM = 1
      OK = .TRUE.
    CASE('n','N')
      IYSYM = 0
      OK = .TRUE.
    CASE DEFAULT
    END SELECT
    SELECT CASE(IARROW)
      CASE(1:9)
        SELECT CASE(YESNO)
        CASE('Y','y','N','n')
          OK = .TRUE.
        CASE DEFAULT
          OK = .FALSE.
        END SELECT
      CASE DEFAULT
        OK = .FALSE.
    END SELECT
  END DO
  Field(12) = YESNO
```

```
        IF (ch(1:1) .NE. YESNO) THEN
          CHANGES = .TRUE.
          ICODE(12) = 1
          Field(12) = YESNO
        ELSE
          Field(12) = ch(1:1)
        END IF
        CALL C_JUSTIFY(Field(12),.FALSE.)
        CALL WriteField(row(12),c,Field(12))

        CASE(13)
        OK = .FALSE.
        DO WHILE (.NOT. OK)
          YESNO = Field(13)(1:1)
          ch(1:1) = YESNO
          CALL C_GETSTR(row(13),col,YESNO,1,.TRUE.,.FALSE.,.FALSE.,
     &            RR,II,IERR,Escape,IARROW)
          SELECT CASE (YESNO(1:1))
          CASE('y','Y')
            SAVE = .TRUE.
            OK = .TRUE.
          CASE('n','N')
            SAVE = .FALSE.
            OK = .TRUE.
          CASE DEFAULT
          END SELECT
          SELECT CASE(IARROW)
            CASE(1:9)
              SELECT CASE(YESNO)
              CASE('Y','y','N','n')
                OK = .TRUE.
              CASE DEFAULT
                OK = .FALSE.
              END SELECT
            CASE DEFAULT
              OK = .FALSE.
          END SELECT
        END DO
        IF (ch(1:1) .NE. YESNO) THEN
          Field(13) = YESNO
        ELSE
          Field(13) = ch(1:1)
        END IF
        CALL C_JUSTIFY(Field(13),.FALSE.)
        CALL WriteField(row(13),c,Field(13))

        IF (SAVE .AND. Field(13)(1:1) .EQ.'Y') THEN
          IF (Fexists) THEN
            CHANGES = .FALSE.
            CALL ShowTires(NTIRES,X,Y,PRESS,RADIUS,
     &                     SAMEp,SAMEr,Fexists,CHANGES)
            IF (CHANGES) ICODE(13) = 1
            Field(13) = 'N'
            CALL WriteField(row(13),c,Field(13))
          ELSE
```

```
      CALL GetTires(X,Y,PRESS,RADIUS,1,NTIRES,
&                  SAMEp,SAMEr,Fexists,CHANGES)
      Field(13) = 'N'
      CALL WriteField(row(13),c,Field(13))
      ICODE(13) = 1
    END IF
    CALL DrawForm(jr,jc,FNAME)
  END IF
  c = jc + 1
  DO k=1,NItems
    CALL WriteField(row(k),c,Field(k))
  END DO


  CASE(14)
  OK = .FALSE.
  DO WHILE (.NOT. OK)
    FNUM7 = Field(14)(1:7)
    RR = RCUT
    RN = RR
    CALL C_GETSTR(row(14),col,FNUM7,7,.FALSE.,.TRUE.,.TRUE.,
&                 RR,II,IERR,Escape,IARROW)
    IF (IERR.EQ.0) THEN
      IF (RR.GE.1 .AND. RR.LE.1000.0) THEN
        OK = .TRUE.
        RCUT = RR
        Field(14)(1:10) = FNUM7
        ICODE(14) = 1
      END IF
    END IF
    SELECT CASE(IARROW)
      CASE(1:9)
        OK = .TRUE.
      CASE DEFAULT
        OK = .FALSE.
    END SELECT
  END DO
  CALL WriteField(row(14),c,Field(14))
  IF (RN .NE. RCUT) CHANGES = .TRUE.

  CASE(15)
  OK = .FALSE.
  SAVE = .FALSE.
  DO WHILE (.NOT. OK)
    YESNO = 'N'
    ch(1:1) = YESNO
    CALL C_GETSTR(row(15),col,YESNO,1,.TRUE.,.FALSE.,.FALSE.,
&                 RR,II,IERR,Escape,IARROW)
    SELECT CASE (YESNO(1:1))
    CASE('y','Y')
      SAVE = .TRUE.
      OK = .TRUE.
      DO k=1, 14
        IF (ICODE(k).EQ.0) THEN
          SAVE = .FALSE.
```

```
            YESNO = 'N'
            EXIT
          END IF
        END DO
      CASE('n','N')
        SAVE = .FALSE.
        OK = .TRUE.
      CASE DEFAULT
      END SELECT
      SELECT CASE(IARROW)
        CASE(1:9)
          SELECT CASE(YESNO)
          CASE('Y','y','N','n')
            OK = .TRUE.
          CASE DEFAULT
            OK = .FALSE.
          END SELECT
        CASE DEFAULT
          OK = .FALSE.
      END SELECT
    END DO
    IF (ch(1:1) .NE. YESNO) THEN
      Field(15) = YESNO
    ELSE
      Field(15) = ch(1:1)
    END IF
    CALL C_JUSTIFY(Field(15),.FALSE.)
    CALL WriteField(row(15),c,Field(15))

    IF (SAVE) THEN
      CALL SaveData(FNAME,NTIRES,X,Y,PRESS,RADIUS,RESW,NDEPTHS,
   &                DZ,GDX,GDY,IXSYM,IYSYM,RCUT)
      CALL GetESWLF(FNAME,NDEPTHS)
      CH\NGES = .FALSE.
      CA . DrawForm(jr,jc,FNAME)
      c = jc + 1
      DO k=1,NItems
        CALL WriteField(row(k),c,Field(k))
      END DO
    END IF

    CASE(16)
    OK = .FALSE.
    SAVE = .FALSE.
    Field(16) = 'N'
    DO WHILE (.NOT. OK)
      YESNO = Field(16)(1:1)
      ch(1:1) = YESNO
      CALL C_GETSTR(row(16),col,YESNO,1,.TRUE.,.FALSE.,.FALSE.,
   &              RR,II,IERR,Escape,IARROW)
      SELECT CASE (YESNO(1:1))
      CASE('y','Y')
       IF (CHANGES) THEN
        SAVE = .FALSE.
        YESNO = 'N'
```

```
          CALL ReCalc
          CALL DrawForm(jr,jc,FNAME)
          c = jc + 1
          DO k=1,NItems
            CALL WriteField(row(k),c,Field(k))
          END DO
         ELSE
          SAVE = .TRUE.
          OK = .TRUE.
         END IF
        CASE('n','N')
          SAVE = .FALSE.
          OK = .TRUE.
        CASE DEFAULT
        END SELECT
        SELECT CASE(IARROW)
          CASE(1:9)
            SELECT CASE(YESNO)
            CASE('Y','y','N','n')
              OK = .TRUE.
            CASE DEFAULT
              OK = .FALSE.
            END SELECT
          CASE DEFAULT
            OK = .FALSE.
        END SELECT
      END DO
      IF (ch(1:1) .NE. YESNO) THEN
        Field(16) = YESNO
      ELSE
        Field(16) = ch(1:1)
      END IF
      CALL C_JUSTIFY(Field(16),.FALSE.)
      CALL WriteField(row(16),c,Field(16))

      IF (SAVE) THEN
        OUTFILE = 'ESWLF.OUT'
        CALL ShowRes(FNAME,OUTFILE)
        CALL DrawForm(jr,jc,FNAME)
        c = jc + 1
        DO k=1,NItems
          CALL WriteField(row(k),c,Field(k))
        END DO
      END IF

      CASE(17)
      OK = .FALSE.
      SAVE = .FALSE.
      Field(17) = 'N'
      DO WHILE (.NOT. OK)
        YESNO = Field(17)(1:1)
        ch(1:1) = YESNO
        CALL C_GETSTR(row(17),col,YESNO,1,.TRUE.,.FALSE.,.FALSE.,
     &             RR,II,IERR,Escape,IARROW)
        SELECT CASE (YESNO(1:1))
```

```
    CASE('y','Y')
      SAVE = .TRUE.
      OK = .TRUE.
    CASE('n','N')
      SAVE = .FALSE.
      OK = .TRUE.
    CASE DEFAULT
    END SELECT
    SELECT CASE(IARROW)
      CASE(1:9)
        SELECT CASE(YESNO)
        CASE('Y','y','N','n')
          OK = .TRUE.
        CASE DEFAULT
          OK = .FALSE.
        END SELECT
      CASE DEFAULT
        OK = .FALSE.
    END SELECT
  END DO
  IF (ch(1:1) .NE. YESNO) THEN
    Field(17) = YESNO
  ELSE
    Field(17) = ch(1:1)
  END IF
  CALL C_JUSTIFY(Field(17),.FALSE.)
  CALL WriteField(row(17),c,Field(17))

  IF (SAVE) THEN
    QUIT = .TRUE.
  END IF

  END SELECT   !OF ITEMS

  SELECT CASE(IARROW)
   CASE(4,8)             !up & left arrow keys
     IF(ITEM.EQ.1) THEN
       ITEM = 17
     ELSE
       ITEM = ITEM - 1
     END IF
   CASE(2,5,6)           !down & right arrow or ENTER keys
     IF (ITEM.EQ.17) THEN
       ITEM = 1
     ELSE
       ITEM = ITEM + 1
     END IF
   CASE(7,9)           !Home or PgUp keys
     ITEM = 1
   CASE(1,3)
     ITEM = 17         !End or PgDn keys
   CASE(0)             !Esc key
                       !don't move, reset field
   CASE DEFAULT
  END SELECT
```

```
      END DO        !CYCLE UNTIL EXIT PROGRAM

      CALL ScreenBk

      RETURN
      END
C*********************************************************************
      SUBROUTINE DrawForm(ir,ic,FNAME)

      CHARACTER*78 BA(17), Title
      CHARACTER FNAME*12

      Title = 'INPUT PARAMETERS FOR DATA FILE : [ '//FNAME//' ]'
      BA = ' '
      NItems = 17
      BA(1) =' Same Contact Pressure for all tires? (Y/N)úúúúúúúú■'
      BA(2) =' Same Radius of contact area for all tires? (Y/N)úú■'
      BA(3) =' Number of tires in gear (100 Max)úúúúúúúúúúúúúúúúú■'
      BA(4) =' Tires Contact Pressure (psi)úúúúúúúúúúúúúúúúúúúúúú■'
      BA(5) =' Tires Radius of contact area (in)úúúúúúúúúúúúúúúúú■'
      BA(6) =' ESWL Radius of contact area (in)úúúúúúúúúúúúúúúúúú■'
      BA(7) =' Number of computational depths (100 Max)úúúúúúúúúú■'
      BA(8) =' Depth Increment (in)úúúúúúúúúúúúúúúúúúúúúúúúúúúúúú■'
      BA(9) =' Grid X-Increment? (in)úúúúúúúúúúúúúúúúúúúúúúúúúúúú■'
      BA(10)=' Grid Y-increment? (in)úúúúúúúúúúúúúúúúúúúúúúúúúúúú■'
      BA(11)=' Symmetry along X-Axis? (Y/N)úúúúúúúúúúúúúúúúúúúúúú■'
      BA(12)=' Symmetry along Y-Axis? (Y/N)úúúúúúúúúúúúúúúúúúúúúú■'
      BA(13)=' Enter or change Tire Coordinates? (Y/N)úúúúúúúúúúú■'
      BA(14)=' Deflection Cutoff in radii (>=100 No Cutoff)úúúúúú■'
      BA(15)=' Proceed with Calculations? (Y/N)úúúúúúúúúúúúúúúúúú■'
      BA(16)=' Display and Print Results? (Y/N)úúúúúúúúúúúúúúúúúú■'
      BA(17)=' Back to Main Menu? (Y/N) úúúúúúúúúúúúúúúúúúúúúúúúú■'

      CALL C_CBOX(Nitems,BA,Title,irow,icol,10)
      ir = irow
      ic = icol

      RETURN
      END
C*********************************************************************
      SUBROUTINE GetTires(x,y,PRESS,RADIUS,is,ie,
     &                    SAMEp,SAMEr,Fexists,CHANGES)

      LOGICAL SAMEp, SAMEr, Fexists, CHANGES

      LOGICAL Ok
      INTEGER row(20), COL

      CHARACTER INTF*3, cn*3,INTF10*10
      CHARACTER BA(5)*78
      CHARACTER Title*78

      INTEGER r,c,jr,jc
      LOGICAL SAVE
```

```
DIMENSION X(100), Y(100), PRESS(100), RADIUS(100)
CHARACTER*10 Field(20)

LOGICAL QUIT, Escape
CHARACTER FNUM7*7, YESNO*1
INTEGER*2 ITEM

BA = ' '
NItems = 5
BA(1) =' X-Coord. (in)úúúúúúúúúúúúúúúúúúúúú*'
BA(2) =' Y-Coord. (in)úúúúúúúúúúúúúúúúúúúúú*'
IF (.NOT.SAMEp) THEN
  BA(3) =' Contact Pressure (psi)úúúúúúúúúúú*'
END IF
IF (.NOT.SAMEr) THEN
  BA(4) =' Radius of contact area (sq in)úúú*'
END IF
BA(5) ='      Ok ? (Y/N)úúúúúúúúúúúúúúúúúú*'

DO i=is, ie
  WRITE(INTF,'(I3)') i
  READ(INTF,'(A3)') cn
  Title ='COORDINATES OF TIRE NUMBER '//cn
  CALL C_CBOX(Nitems,BA,Title,ir,ic,10)
  jr = ir
  jc = ic
  c = jc+1
  COL = jc
  row(1) = jr
  DO k=2,5
    row(k) = row(k-1) + 1
  END DO
  IF (.NOT.Fexists) THEN
    Field = ' '
  ELSE
    WRITE(INTF10,'(F10.2)') X(i)
    READ(INTF10,'(A10)') Field(1)
    IF (X(i) .EQ. 0.0) Field(1) = '0.0'
    CALL C_JUSTIFY(Field(1),.FALSE.)
    CALL WriteField(row(1),c,Field(1))
    WRITE(INTF10,'(F10.2)') Y(i)
    READ(INTF10,'(A10)') Field(2)
    IF (Y(i) .EQ. 0.0) Field(2) = '0.0'
    CALL C_JUSTIFY(Field(2),.FALSE.)
    CALL WriteField(row(2),c,Field(2))
    IF (.NOT.SAMEp) THEN
      WRITE(INTF10,'(F10.2)') PRESS(i)
      READ(INTF10,'(A10)') Field(3)
      CALL C_JUSTIFY(Field(3),.FALSE.)
      CALL WriteField(row(3),c,Field(3))
    END IF
    IF (.NOT.SAMEr) THEN
      WRITE(INTF10,'(F10.2)') RADIUS(i)
      READ(INTF10,'(A10)') Field(4)
      CALL C_JUSTIFY(Field(4),.FALSE.)
```

```
          CALL WriteField(row(4),c,Field(4))
        END IF
        IF (Fexists) THEN
          Field(5) = 'Y'
        ELSE
          Field(5) = 'N'
        END IF
      END IF

      QUIT = .FALSE.
      IF (Fexists) THEN
        ITEM = 5
      ELSE
        ITEM = 1
      END IF
      DO WHILE (.NOT.QUIT)
      SELECT CASE(ITEM)

      CASE(1)
99    r = jr
      c = jc+1
      OK = .FALSE.
      DO WHILE (.NOT. OK)
        FNUM7 = Field(1)(1:7)
        RR = X(i)
        CALL C_GETSTR(row(1),COL,FNUM7,7,.FALSE.,.TRUE.,.TRUE.,
     &                RR,II,IERR,Escape,IARROW)
        IF (IERR.EQ.0) THEN
          IF ((RR.GE.-9999.) .AND. (RR.LE.9999.)) THEN
            OK = .TRUE.
            X(i) = RR
            Field(1)(1:10) = FNUM7
            CALL WriteField(row(1),c,Field(1))
            CHANGES = .TRUE.
          END IF
        END IF
       END DO

      CASE(2)
      OK = .FALSE.
      DO WHILE (.NOT. OK)
        FNUM7 = Field(2)(1:7)
        RR = Y(i)
        CALL C_GETSTR(row(2),COL,FNUM7,7,.FALSE.,.TRUE.,.TRUE.,
     &                RR,II,IERR,Escape,IARROW)
        IF (IERR.EQ.0) THEN
          IF ((RR.GE.-9999.) .AND. (RR.LE.9999.)) THEN
            OK = .TRUE.
            Y(i) = RR
            Field(2)(1:10) = FNUM7
            CALL WriteField(row(2),c,Field(2))
            CHANGES = .TRUE.
          END IF
        END IF
      END DO
```

```
      CASE(3)
      IF(.NOT.SAMEp) THEN
      OK = .FALSE.
      DO WHILE (.NOT. OK)
        FNUM7 = Field(3)(1:7)
        RR = PRESS(i)
        CALL C_GETSTR(row(3),COL,FNUM7,7,.FALSE.,.TRUE.,.TRUE.,
     &            RR,II,IERR,Escape,IARROW)
        IF (IERR.EQ.0) THEN
          IF ((RR .GE. 0.1) .AND. (RR.LE.1000.)) THEN
            OK = .TRUE.
            PRESS(i) = RR
            Field(3)(1:10) = FNUM7
            CALL WriteField(row(3),c,Field(3))
            CHANGES = .TRUE.
          END IF
        END IF
      END DO
      END IF

      CASE(4)
      IF (.NOT.SAMEr) THEN
      OK = .FALSE.
      DO WHILE (.NOT. OK)
        FNUM7 = Field(4)(1:7)
        RR = RADIUS(i)
        CALL C_GETSTR(row(4),COL,FNUM7,7,.FALSE.,.TRUE.,.TRUE.,
     &            RR,II,IERR,Escape,IARROW)
        IF (IERR.EQ.0) THEN
          IF ((RR.GE. 0.1) .AND. (RR.LE.1000.)) THEN
            OK = .TRUE.
            RADIUS(i) = RR
            Field(4)(1:10) = FNUM7
            CALL WriteField(row(4),c,Field(4))
            CHANGES = .TRUE.
          END IF
        END IF
      END DO
      END IF

      CASE(5)
98    OK = .FALSE.
      Field(5) = 'N'
      DO WHILE (.NOT. OK)
        YESNO = Field(5)(1:1)
        CALL C_GETSTR(row(5),COL,YESNO,1,.TRUE.,.FALSE.,.FALSE.,
     &            RR,II,IERR,Escape,IARROW)
        SELECT CASE (YESNO(1:1))
        CASE('y','Y')
          SAVE = .TRUE.
          QUIT = .TRUE.
          OK = .TRUE.
        CASE('n','N')
          SAVE = .FALSE.
```

```
        OK = .TRUE.
      END SELECT
    END DO
    CALL WriteField(row(5),c,Field(5))

  END SELECT

  SELECT CASE(IARROW)
   CASE(8)               !up arrow key
     IF(ITEM.EQ.1) THEN
       ITEM = 5
     ELSE
       SELECT CASE(ITEM)
       CASE(5)
         IF (SAMEr) THEN
           ITEM = ITEM - 1
           IF (SAMEp) ITEM = ITEM - 1
         END IF
       CASE(4)
         IF (SAMEp) ITEM = ITEM - 1
       END SELECT
       ITEM = ITEM - 1
     END IF

   CASE(2,5)             !down arrow or ENTER keys
     IF (ITEM.EQ.5) THEN
       ITEM = 1
     ELSE
       SELECT CASE(ITEM)
       CASE(2)
         IF (SAMEp) THEN
           ITEM = ITEM + 1
           IF (SAMEr) ITEM = ITEM + 1
         END IF
       CASE(3)
         IF (SAMEr) ITEM = ITEM + 1
       END SELECT
       ITEM = ITEM + 1
     END IF
   CASE(7,9)             !Home or PgUp keys
     ITEM = 1
   CASE(1,3)
     ITEM = 5            !End or PgDn keys
   CASE(0)               !Esc key
                         !don't move, reset field
   CASE DEFAULT
  END SELECT

  END DO    !QUIT LOOP

  END DO    !i LOOP

  Fexists = .TRUE.

  RETURN
```

```
      END
C*********************************************************************
      SUBROUTINE WriteField(ir,ic,strng)

      INCLUDE 'FGRAPH.FD'

      CHARACTER*(*) strng
      RECORD /rccoord/ s

      INTEGER*2 dummy2,fg
      INTEGER*4 dummy4,bk

      fg = GetTextColor()
      bk = GetBkColor()
      dummy2 = SetTextColor(1)  !blue
      dummy4 = SetBkColor(7)    !white

      CALL C_CURSOROFF
      CALL SetTextPosition(ir,ic,s)
      CALL OutText(strng)
      CALL SetTextPosition(ir,ic,s)
      CALL C_CURSORON

      dummy2 = SetTextColor(fg)
      dummy4 = SetBkColor(bk)

      RETURN
      END
C*********************************************************************
      SUBROUTINE ReadFile(FNAME,NTIRES,X,Y,PRESS,RADIUS,RESW,NDEPTHS,
     &                    DZ,GDX,GDY,IXSYM,IYSYM,RCUT,Fexists)

      CHARACTER FNAME*12
      DIMENSION X(100), Y(100), PRESS(100), RADIUS(100)
      CHARACTER CDUMMY*80
      LOGICAL Fexists

      INQUIRE(FILE=FNAME,EXIST=Fexists)
      IF (.NOT.Fexists) RETURN

      CALL C_OUNIT(IDISK)
      OPEN(IDISK,FILE=FNAME)
        READ(IDISK,'(A)') CDUMMY
        READ(IDISK,*) NTIRES
        IF (NTIRES. GT. 100) THEN
          STOP 'ERROR: Maximum number of tires should be <= 100.'
        END IF
        READ(IDISK,'(A)') CDUMMY
        DO i=1, NTIRES
          READ(IDISK,*) X(i), Y(i), PRESS(i), RADIUS(i)
        END DO
        READ(IDISK,'(A)') CDUMMY
        READ(IDISK,*) RESW
        READ(IDISK,'(A)') CDUMMY
        READ(IDISK,*) NDEPTHS
```

```
        READ(IDISK,'(A)') CDUMMY
        READ(IDISK,*) DZ
        IF (NDEPTHS .GT. 100) THEN
          STOP ' ERROR: Maximum number of depths should be <= 100.'
        END IF
        READ(IDISK,'(A)') CDUMMY
        READ(IDISK,*) GDX, GDY
        READ(IDISK,'(A)') CDUMMY
        READ(IDISK,*) IXSYM, IYSYM
        READ(IDISK,'(A)') CDUMMY
        READ(IDISK,*) RCUT

      CLOSE(IDISK)
      RETURN
      END
C**********************************************************************
      SUBROUTINE SaveData(FNAME,NTIRES,X,Y,PRESS,RADIUS,RESW,NDEPTHS,
     &                    DZ,GDX,GDY,IXSYM,IYSYM,RCUT)
      CHARACTER FNAME*12
      DIMENSION X(100), Y(100), PRESS(100), RADIUS(100)
      CHARACTER S(20)*80

      S(1)='Number of tires in gear'
      S(2)='(X,Y) tire coordinates, pressure, radius of contact area'
      S(3)='ESWL Radius of contact area'
      S(4)='Number of depths ( if = 1 , ESWLF calculated at DZ)'
      S(5)='Depth increment (DZ)'
      S(6)='DX, DY grid increments'
      S(7)='Gear symmetry along X & Y gear axes (0 if not symmetric)'
      S(8)='Deflection Cutoff Distance in radii (if >= 100 no cutoff)'

      CALL C_OUNIT(IDISK)
      OPEN(IDISK,FILE=FNAME)

      WRITE(IDISK,'(A)') S(1)
      WRITE(IDISK,'(I3)') NTIRES
      WRITE(IDISK,'(A)') S(2)
      DO i=1, NTIRES
        WRITE(IDISK,'(4(F10.2))') X(i), Y(i), PRESS(i), RADIUS(i)
      END DO
      WRITE(IDISK,'(A)') S(3)
      WRITE(IDISK,'(F10.2)') RESW
      WRITE(IDISK,'(A)') S(4)
      WRITE(IDISK,'(I3)') NDEPTHS
      WRITE(IDISK,'(A)') S(5)
      WRITE(IDISK,'(F10.2)') DZ
      WRITE(IDISK,'(A)') S(6)
      WRITE(IDISK,'(2(F10.2))') GDX,GDY
      WRITE(IDISK,'(A)') S(7)
      WRITE(IDISK,'(2I3)') IXSYM, IYSYM
      WRITE(IDISK,'(A)') S(8)
      WRITE(IDISK,'(F10.2)') RCUT

      CLOSE(IDISK)
```

```
      RETURN
      END
C*********************************************************************
      SUBROUTINE ShowRes(FNAME,OUTFILE)

      CHARACTER*12 FNAME, OUTFILE, INFI E
      LOGICAL exists
      LOGICAL OK, GOPRINT
      CHARACTER*78 LA(100), Title, BA(5)
      CHARACTER*80 tp, MSG, CDUMMY

      INQUIRE(FILE=OUTFILE,EXIST=exists)
      IF(.NOT.exists) THEN
        CALL NoResults
        RETURN
      END IF

      CAL. `_OUNIT(IDISK)
      OPEN(.DISK,FILE=OUTFILE)
      READ(IDISK,'(30X,A12)') INFILE
      IF (FNAME .NE. ' ') THEN
        IF (INFILE.NE.FNAME) THEN
          Nitems = 5
          Title = 'CANNOT PERFORM REQUESTED OPERATION!'
          BA = ' '
          BA(1) = ' The existing output data file does not match'
          BA(2) = ' the data in the input data file.  Re-calculate'
          BA(3) = ' the ESWLF factors.'
          BA(5) = '            Press any key to continue.'
          CALL C_CURSOROFF
          CALL C_CBOXFLASH(N .ems,BA,Title)
          CALL C_WAITFORKEY
          CALL C_CURSORON
          CLOSE(IDISK)
          RETURN
        END IF
      END IF
      READ(IDISK,'(A)') Title
      READ(IDISK,'(A)') CDUMMY
      DO j=1, 100       !max # depths = 100
        READ(IDISK,'(A)',END=10) LA(j)
      END DO
10    Nitems = j - 1
      CLOSE(IDISK)
      max = 15   !max depths/window
      IF (Nitems.LT.max) max = Nitems
      Longest = 1
      DO i=1, Nitems
        IF (LEN_TRIM(LA(i)).GT.Longest) Longest = LEN_TRIM(LA(i))
      END DO
      IF (LEN_TRIM(Title).GT.Longest) Longest = LEN_TRIM(Title)
      icol = 1+(80-Longest-2)/2
      irow = 1+(25-max-4)/2

      TP = 'ESWL FACTOR RESULTS'
```

```
      MSG='[** PgUp  PgDn ] to view,  [ENTER] to print,'//
     &    '  [Esc] back to data entry screen'
      CALL C_SCREENBK(TP,MSG,.FALSE.)
      OK = .FALSE.
      DO WHILE(.NOT.OK)
C        CALL C_LIST(irow,icol,Nitems,max,LA,Title,iPick,Longest)
        CALL C_SLIST(irow,icol,Nitems,max,LA,Title,iPick)
        SELECT CASE(iPick)
          CASE(1:100)   !Print,   100=max num. depths
            OK = .TRUE.
            CALL PrintRes(OUTFILE,GOPRINT)
            IF (.NOT.GOPRINT) OK = .FALSE.
          CASE(0)       !Esc
            OK = .TRUE.
          CASE DEFAULT
        END SELECT
        CALL C_SCREENBK(TP,MSG,.FALSE.)
      END DO
      CALL ScreenBk

      RETURN
      END
C***********************************************************************
      SUBROUTINE ShowTires(NTIRES,X,Y,PRESS,RADIUS,
     &                     SAMEp,SAMEr,Fexists,CHANGES)

C      INCLUDE 'TOOLS.STR'
      CHARACTER LA(100)*78, Title*78
      CHARACTER*80 TP, MSG
      CHARACTER*12 S1, S2, S3, S4, S5
      LOGICAL SAMEp, SAMEr, Fexists, CHANGES, OK
      DIMENSION X(100), Y(100), PRESS(100), RADIUS(100)

      Nitems = NTIRES
C             1          12          24          36          48          60
      Title = '  Tire #    X-coord(in)  Y-coord(in)'//
     &        '  Radius(in)  Press(psi)'

      DO i=1, Nitems
        WRITE(S1,'(I12)')   i
        WRITE(S2,'(F12.2)') X(i)
        WRITE(S3,'(F12.2)') Y(i)
        WRITE(S4,'(F12.2)') RADIUS(i)
        WRITE(S5,'(F12.2)') PRESS(i)
        LA(i) = S1//S2//S3//S4//S5
      END DO
      max = 15   !max depths/window
      IF (Nitems.LT.max) max = Nitems
      Longest = 1
      DO i=1, Nitems
        IF (LEN_TRIM(LA(i)).GT.Longest) Longest = LEN_TRIM(LA(i))
      END DO
      IF (LEN_TRIM(Title).GT.Longest) Longest = LEN_TRIM(Title)
      icol = 1+(80-Longest-2)/2
      irow = 1+(25-max-4)/2
```

```
      TP = 'SELECT TIRE TO CHANGE'
      MSG='[** PgUp PgDn ] to mark, [ENTER] to select,'//
     &    ' [Esc] when done'
      OK = .FALSE.
      DO WHILE(.NOT.OK)
        DO i=1, Nitems
          WRITE(S1,'(I10)')   i
          WRITE(S2,'(F10.2)') X(i)
          WRITE(S3,'(F12.2)') Y(i)
          WRITE(S4,'(F12.2)') RADIUS(i)
          WRITE(S5,'(F12.2)') PRESS(i)
          LA(i) = S1//S2//S3//S4//S5
        END DO
        CALL C_SCREENBK(TP,MSG,.FALSE.)
C        CALL C_LIST(irow,icol,Nitems,max,LA,Title,iPick,Longest)
        CALL C_SLIST(irow,icol,Nitems,max,LA,Title,iPick)
        SELECT CASE(iPick)
          CASE(1:100)       !100=max num. of tires
          CALL GetTires(X,Y,PRESS,RADIUS,iPick,iPick,
     &                  SAMEp,SAMEr,Fexists,CHANGES)
          CASE(0)           !Esc
            OK = .TRUE.
          CASE DEFAULT
        END SELECT
      END DO
      CALL ScreenBk

      RETURN
      END
C**********************************************************************
      SUBROUTINE NoResults

      CHARACTER BA(6)*78, Title*78

      Title = 'OUTPUT DATA FILE DOES NOT EXISTS!'
      BA = ' '
      NItems = 6
      BA(1) = ' No output data file has been generated.'
      BA(2) = ' Please, make sure to enter your data and'
      BA(3) = ' calculate the ESWL Factors before selecting'
      BA(4) = ' this option.'
      BA(6) = '      Press any key to continue.'
      CALL C_CURSOROFF
      CALL C_CBOXFLASH(Nitems,BA,Title)
      CALL C_WAITFORKEY
      CALL C_CURSORON
      CALL ScreenBk

      RETURN
      END
C**********************************************************************
      SUBROUTINE ReCalc

      CHARACTER BA(4)*78, Title*78
```

```
      BA = ' '
      Title = 'PERFORM CALCULATIONS FIRST!'
      NItems = 4
      BA(1) = ' Either your data was changed or no'
      BA(2) = ' ESWLF calculations have been performed.'
      BA(4) = '      Press any key to continue.'
      CALL C_CURSOROFF
      CALL C_CBOXFLASH(Nitems,BA,Title)
      CALL C_WAITFORKEY
      CALL C_CURSORON
      CALL ScreenBk

      RETURN
      END
C**********************************************************************
      SUBROUTINE PrintRes(OUTFILE,OkToPrint)

      CHARACTER OUTFILE*12, INFILE*12
      LOGICAL OK, OkToPrint
      CHARACTER DUMMY*80
      DIMENSION X(100), Y(100), PRESS(100), RADIUS(100)
      CHARACTER BA(3)*78, Title*78

      BA = ' '
      NItems = 3
      Title = 'ABOUT TO PRINT RESULTS'
      BA(1) = '  Please, verify the printer is ready.'
      BA(3) = ' Press [ENTER] to print, [Esc] to CANCEL'

      CALL C_CURSOROFF
      OK = .FALSE.
      OkToPrint = .FALSE.
      DO WHILE (.NOT.OK)
        CALL C_CBOXFLASH(Nitems,BA,Title)
        CALL C_GETASC(IKEY)
        SELECT CASE(IKEY)
          CASE(13)         ! ENTER
            OK = .TRUE.
            OkToPrint = .TRUE.
          CASE(27)         ! Esc
            OK = .TRUE.
            OkToPrint = .FALSE.
          CASE DEFAULT
        END SELECT
      END DO
      IF (OkToPrint) THEN
C        CALL C_OUNIT(LU)
        IDISK = 3
C        CALL C_OUNIT(LU)
        IPRNT = 4
        OPEN(IDISK,FILE=OUTFILE)
          READ(IDISK,'(30X,A12)') INFILE
        CLOSE(IDISK)
        OPEN(IDISK,FILE=INFILE)
```

```
      OPEN(IPRNT,FILE='LPT1')
         WRITE(IPRNT,100)   !form feed
100      FORMAT('1')
         WRITE(IPRNT,'(1X,A)') 'INPUT PARAMETERS'
        READ(IDISK,'(A)') DUMMY
         WRITE(IPRNT,'(1X,A)') DUMMY
        READ(IDISK,'(I3)') NTIRES
         WRITE(IPRNT,'(1X,I3)') NTIRES
        READ(IDISK,'(A)') DUMMY
         WRITE(IPRNT,'(1X,A)') DUMMY
        DO i=1, NTIRES
          READ(IDISK,'(4(F10.2))') X(i), Y(i),
     &                   PRESS(i), RADIUS(i)
           WRITE(IPRNT,'(1X,4(F10.2))') X(i), Y(i),
     &                   PRESS(i), RADIUS(i)
        END DO
        READ(IDISK,'(A)') DUMMY
         WRITE(IPRNT,'(1X,A)') DUMMY
        READ(IDISK,'(F10.2)') RESW
         WRITE(IPRNT,'(1X,F10.2)') RESW
        READ(IDISK,'(A)') DUMMY
         WRITE(IPRNT,'(1X,A)') DUMMY
        READ(IDISK,'(I3)') NDEPTHS
         WRITE(IPRNT,'(1X,I3)') NDEPTHS
        READ(IDISK,'(A)') DUMMY
         WRITE(IPRNT,'(1X,A)') DUMMY
        READ(IDISK,'(F10.2)') DZ
         WRITE(IPRNT,'(1X,F10.2)') DZ
        READ(IDISK,'(A)') DUMMY
         WRITE(IPRNT,'(1X,A)') DUMMY
        READ(IDISK,'(2(F10.2))') GDX,GDY
         WRITE(IPRNT,'(1X,2(F10.2))') GDX,GDY
        READ(IDISK,'(A)') DUMMY
         WRITE(IPRNT,'(1X,A)') DUMMY
        READ(IDISK,'(2I3)') IXSYM, IYSYM
         WRITE(IPRNT,'(1X,2I3)') IXSYM, IYSYM
        READ(IDISK,'(A)') DUMMY
         WRITE(IPRNT,'(1X,A)') DUMMY
        READ(IDISK,'(A)') DUMMY
         WRITE(IPRNT,'(1X,A)') DUMMY
      CLOSE(IDISK)
      OPEN(IDISK,FILE=OUTFILE)
        WRITE(IPRNT,'(1X,A)') ' '
      READ(IDISK,'(A)') DUMMY
      READ(IDISK,'(A)') DUMMY
       WRITE(IPRNT,'(1X,A)') DUMMY
      READ(IDISK,'(A)') DUMMY
       WRITE(IPRNT,'(1X,A)') DUMMY
      DO i=1, NDEPTHS
        READ(IDISK,'(A)') DUMMY
        WRITE(IPRNT,'(1X,A)') DUMMY
      END DO
      CLOSE(IDISK)
      CLOSE(IPRNT)
    END IF
```

```
      CALL C_CURSORON
      RETURN
      END
C***********************************************************************
      SUBROUTINE GetESWLF(FNAME,NDEPTHS)

      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER FNAME*12
      DIMENSION X(100), Y(100), PRESS(100), RADIUS(100), Z(100)
      DIMENSION DEFMAX(100), XMAXD(100), YMAXD(100)
      DIMENSION ESWL(100), ESWLF(100)
      CHARACTER CDUMMY*1
      LOGICAL EXISTS

      CALL C_OUNIT(LU)
      IDISK = LU
C     CALL C_OUNIT(LU)
C     IFILE = LU
      INQUIRE(FILE=FNAME,EXIST=EXISTS)
      IF (.NOT. EXISTS) THEN
        STOP ' ERROR: Data file does not exists.'
      END IF
C     IF (NDEPTHS .EQ. 1) THEN
C       OPEN(IFILE,FILE='DEFL.OUT')
C     END IF
      OPEN(IDISK,FILE=FNAME)
        READ(IDISK,'(A)') CDUMMY
        READ(IDISK,*) NTIRES
        IF (NTIRES. GT. 100) THEN
          STOP 'ERROR: Maximum number of tires should be <= 100.'
        END IF
        READ(IDISK,'(A)') CDUMMY
        DO i=1, NTIRES
          READ(IDISK,*) X(i), Y(i), PRESS(i), RADIUS(i)
        END DO
        READ(IDISK,'(A)') CDUMMY
        READ(IDISK,*) RESW
        READ(IDISK,'(A)') CDUMMY
        READ(IDISK,*) NDEPTHS
        READ(IDISK,'(A)') CDUMMY
        READ(IDISK,*) DZ
        IF (NDEPTHS .GT. 100) THEN
          STOP ' ERROR: Maximum number of depths should be <= 100.'
        END IF
        IF (DZ .EQ. 0.0) NDEPTHS = 1
        IF (NDEPTHS.EQ.1) THEN
          Z(1) = DZ
        END IF
        DO i=2, NDEPTHS
          Z(i) = Z(i-1) + DZ
        END DO
        READ(IDISK,'(A)') CDUMMY
        READ(IDISK,*) GDX, GDY
        READ(IDISK,'(A)') CDUMMY
```

```
      READ(IDISK,*) IXSYM, IYSYM
      READ(IDISK,'(A)') CDUMMY
      READ(IDISK,*) RCUT

    CLOSE(IDISK)

    IF (GDX.EQ.0.0 .OR. GDY.EQ.0.0) THEN
      STOP ' ERROR: Neither GDX or GDY can be equal to zero.'
    END IF

    CALL StatusBox
    CALL C_CURSOROFF

    PI = 3.141592653589793238
    XMIN =  999999.0
    YMIN =  999999.0
    XMAX = -999999.0
    YMAX = -999999.0
    DO i=1, NTIRES
      IF (X(i) .LT. XMIN) XMIN = X(i)
      IF (Y(i) .LT. YMIN) YMIN = Y(i)
      IF (X(i) .GT. XMAX) XMAX = X(i)
      IF (Y(i) .GT. YMAX) YMAX = Y(i)
    END DO
C     XMIN = XMIN - 2.0*RADIUS(1)
C     YMIN = YMIN - 2.0*RADIUS(1)
C     XMAX = XMAX + 2.0*RADIUS(1)
C     YMAX = YMAX + 2.0*RADIUS(1)
    GX0 = XMIN
    GY0 = YMIN

    IF (IYSYM .EQ. 1) THEN
      XN = ((XMAX - XMIN)/2.0)/GDX
    ELSE
      XN = (XMAX - XMIN)/GDX
    END IF
    IF (IXSYM .EQ. 1) THEN
      YN = ((YMAX - YMIN)/2.0)/GDY
    ELSE
      YN = (YMAX - YMIN)/GDY
    ENDIF
    IF ( XN .EQ. INT(XN) ) THEN
      NLX = XN + 1.0
    ELSE
      NLX = XN + 2.0
    END IF
    IF ( YN .EQ. INT(YN) ) THEN
      NLY = YN + 1.0
    ELSE
      NLY = YN + 2.0
    END IF

    DO i=1, NDEPTHS
      DEFMAX(i) = -999999.0
    END DO
```

```
      ITOTAL = NDEPTHS*NLX*NLY*NTIRES
      ICOUNT = 0

      DO KZ=1, NDEPTHS
        GY = GY0
        DO KY=1, NLY
          GX = GX0
          DO KX=1, NLX
            WZSUM = 0.0
            DO KT=1, NTIRES
              ICUT = 0                        !compute defl. at cutoff dist.
              IF (RCUT.GE.100.0) ICUT = 2     !without cutoff distance
88            CONTINUE
              IF (ICUT.EQ.0) THEN
                R0 = RCUT*RADIUS(KT)
                ROFF = R0
              ELSE
                R0 = SQRT( (X(KT)-GX)**2 + (Y(KT)-GY)**2 )
              END IF
              IF (R0.LT.0.000001) R0=0.0
              R1 = SQRT( Z(KZ)**2 + (R0-RADIUS(KT))**2 )
              R2 = SQRT( Z(KZ)**2 + (R0+RADIUS(KT))**2 )
              IF (Z(KZ) .EQ. 0.0) THEN
                IF (R0.EQ.0.0) THEN
                  WZ = (3.0*RADIUS(KT)*PRESS(KT)/2.0)
                  GOTO 99
                END IF
                IF (R0.EQ.RADIUS(KT)) THEN
                  WZ = (3.0*RADIUS(KT)*PRESS(KT)/PI)
                  GOTO 99
                END IF
                IF (R0.GT.RADIUS(KT)) THEN
                  sk = RADIUS(KT)/R0
                  skc = SQRT(1.0-sk**2)
                  skc2 = 1.0-sk**2
                  CE = CEL(skc,1.D0,1.D0,skc2)
                  CK = CEL(skc,1.D0,1.D0,1.D0)
                  XX = CE - (1.0-sk**2)*CK
                  WZ = ((3.0*PRESS(KT)*RADIUS(KT)/PI)*XX)
                  GOTO 99
                END IF
                IF (R0.GT.0.0 .AND. R0.LT.RADIUS(KT)) THEN
                  sk = R0/RADIUS(KT)
                  skc = SQRT(1.0-sk**2)
                  skc2 = 1.0-sk**2
                  CE = CEL(skc,1.D0,1.D0,skc2)
                  WZ = (3.0*RADIUS(KT)*CE*PRESS(KT)/PI)
                END IF
              ELSE
                IF (R0 .EQ. 0.0) THEN
                  WZ = (3.0*PRESS(KT)*(RADIUS(KT)**2)/(2.0*R1))
                ELSE
                  COSV = (Z(KZ)**2 - RADIUS(KT)**2 + R0**2)/(R1*R2)
                  skc = R1/R2
```

```
                skc2 = skc**2
                CE = CEL(skc,1.D0,1.D0,skc2)
                CK = CEL(skc,1.D0,1.D0,1.D0)
                XM = R2*CE
                XN = R1*CK
                WZ = (3.0*PRESS(KT)*(XM-XN*COSV)/(2.0*PI))
              END IF
            END IF
99          CONTINUE
            IF (ICUT.EQ.2) THEN
              ICOUNT = ICOUNT + 1
              CALL STATUS(ICOUNT,ITOTAL)
            ELSE
              IF (ICUT.EQ.0) THEN
                DOFF = WZ
                ICUT = 1
                GOTO 88
              ELSE
                IF (RO.GE.ROFF) THEN
                  WZ = 0.0
                ELSE
                  WZ = WZ - (DOFF*(RO/ROFF))
                END IF
                ICOUNT = ICOUNT + 1
                CALL STATUS(ICOUNT,ITOTAL)
              END IF
            END IF
            WZSUM = WZSUM + WZ
          END DO   !KT

C          IF (NDEPTHS .EQ. 1) THEN
C            WRITE(IFILE,'(3(F10.2))') GX,GY,(-1)*WZ
C          END IF
          IF (WZSUM .GT. DEFMAX(KZ)) THEN
            DEFMAX(KZ) = WZSUM
            XMAXD(KZ) = GX
            YMAXD(KZ) = GY
          END IF
          GX = GX + GDX
        END DO      !KX
        GY = GY + GDY
      END DO        !KX
    END DO          !KZ
C    IF (NDEPTHS .EQ. 1) CLOSE(IFILE)

    GEARL = 0.0
    DO i=1, NTIRES
      GEARL = GEARL + PRESS(i)*PI*(RADIUS(i)**2)
    END DO

    DO i=1, NDEPTHS
      R = SQRT( Z(i)**2 + RESW**2 )
      ESWL(i) = 2.0*R*PI*DEFMAX(i) / 3.0
      ESWLF(i) = ESWL(i) / GEARL
    END DO
```

```
      OPEN(IDISK,FILE='ESWLF.OUT')
        WRITE(IDISK,101) FNAME
        WRITE(IDISK,100)
        DO i=1, NDEPTHS
          WRITE(IDISK,110) Z(i), ESWLF(i), XMAXD(i), YMAXD(i)
C           WRITE(IDISK,110) Z(i),DEFMAX(i),ESWLF(i),XMAXD(i),YMAXD(i)
        END DO
      CLOSE(IDISK)
  101 FORMAT(1X,'RESULTS FOR INPUT DATA FILE: ',A12)
C                         +           +           +           +           +
C                1234567890123456789012345678901234567890123456789012345678901234567890
  100 FORMAT(2X,' DEPTH(in)    ESWLF       XMAX(in)    YMAX(in)',
     &     /,2X,'---------------------------------------------------')
  110 FORMAT(F10.2,2X,F10.3,2X,2(F10.2))

C  100 FORMAT(2X,' DEPTH(in)  WZ*Em(in*psi)  ESWLF',
C     &'       XMAX(in)    YMAX(in)',
C     &     /,2X,'---------------------------------',
C     &'--------------------------')
C  110 FORMAT(2(F10.2,2X),F10.3,2X,2(F10.2))

      CALL C_CURSORON
      RETURN
      END

C***********************************************************************
C     Complete elliptic integral solution (1st and 2nd kind) from:
C     'NUMERICAL RECIPES(FORTRAN)', CAMBRIDGE, REPRINT 1990
C     pp. 187-188
C***********************************************************************

      FUNCTION CEL(QQC,PP,AA,BB)
      IMPLICIT REAL*8 (A-H,O-Z)

      PARAMETER (CA=.000001, PIO2=1.5707963268)

      IF (QQC .EQ. 0.0) THEN
        STOP ' Failure in CEL function.'
      END IF
      QC = ABS(QQC)
      A = AA
      B = BB
      P = PP
      E = QC
      EM = 1.0
      IF (P .GT. 0.0) THEN
         P = SQRT(P)
         B = B/P
      ELSE
         F = QC*QC
         Q = 1.0 - F
         G = 1.0 - P
         F = F - P
         Q = Q*(B-A*P)
```

```
            P = SQRT(F/G)
            A = (A-B) / G
            B = -Q/(G*G*P) + A*P
         END IF
 1       F = A
         A = A + B/P
         G = E/P
         B = B + F*G
         B = B+B
         P = G+P
         G = EM
         EM = QC+EM
         IF (ABS(G-QC) .GT. G*CA) THEN
            QC = SQRT(E)
            QC = QC+QC
            E = QC*EM
            GOTO 1
          END IF
          CEL = PIO2*(B+A*EM)/(EM*(EM+P))

         RETURN
         END
C***********************************************************************
```

Waterways Experiment Station Cataloging-in-Publication Data

Gonzalez, Carlos R.
A computer program for the calculation of the equivalent single-wheel load factor / by Carlos R. Gonzalez ; prepared for Department of the Army, U.S. Army Corps of Engineers and Department of the Air Force, Air Force Engineering and Services Center.
63 p. : ill. ; 28 cm. -- (Miscellaneous paper ; GL-92-18)
Includes bibliographic references.
1. Pavements -- Live loads -- Testing -- Computer programs. 2. Runways (Aeronautics) -- Testing -- Computer programs. 3. Pavements, Flexible -- Testing -- Computer programs. I. United States. Army. Corps of Engineers. II. Air Force Engineering and Services Center (U.S.) III. U.S. Army Engineer Waterways Experiment Station. IV. Title. V. Series: Miscellaneous paper (U.S. Army Engineer Waterways Experiment Station) ; GL-92-18.
TA7 W34m no.GL-92-18